ENSEEIHT 2ième année Département Sciences du Numérique. Année 2018-2019

Intergiciels

Examen session 1

Daniel Hagimont

Durée: 1h30, documents autorisés

*Lire l'ensemble des énoncés avant de commencer à répondre. La **clarté**, la **précision** et la **concision** des réponses, ainsi que leur **présentation matérielle**, seront des éléments importants d'appréciation. Donnez essentiellement les programmes Java demandés. Dans vos programmes, vous n'avez pas à programmer les imports et le traitement des exceptions.*

PROBLÈME I (Sockets)

On veut implanter avec des sockets en Java un système de monitoring d'un ensemble de machines réparties.
Sur chacune de ces machines, on exécute un programme sonde appelé *Probe*. Une sonde récupère périodiquement la charge processeur de la machine et l'envoie à un serveur s'exécutant sur une machine unique avec la séquence de code :

```
while (true) {
    Thread.sleep(1000);
    int load = System.getCPULoad();
    // envoi de la valeur load au serveur
}
```

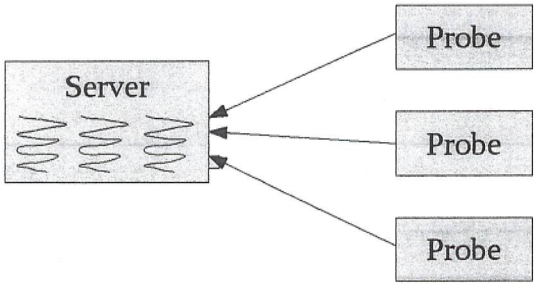
Lorsque la sonde démarre, elle crée une connexion TCP avec le serveur (dont l'adresse IP et le numéro de port sont supposés connus) et utilise cette connexion pour tous ses envois.

Le programme serveur (*Server*) de son coté reçoit les connexions des sondes. Il reçoit les charges processeur des sondes et s'en sert pour calculer la charge moyenne des machines sondées en utilisant une instance de la classe *Average* qui fournit les méthodes :

```
public synchronized void updateOneCPULoad(int load);
        // méthode utilisée par Server pour prendre en compte une charge reçue
public synchronized int getAverageCPULoad();
        // méthode utilisée par une application externe pour récupérer la charge moyenne
```

Le serveur est multi-threadé et la classe *Average* (qui est fournie) est synchronisée.

Donnez l'implantation des programmes *Probe* et *Server*.



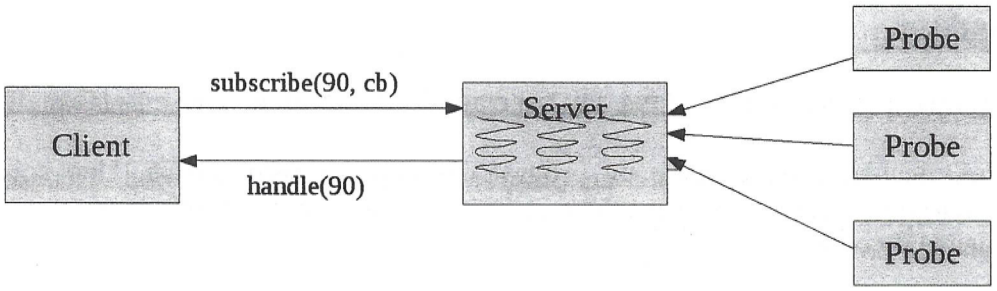
PROBLÈME II (RMI)

On suppose que la méthode *getAverageCPULoad()* retourne un pourcentage multiple de 10. On veut permettre à des programmes clients d'appeler avec RMI le serveur pour s'abonner à un niveau de charge processeur avec la méthode :

public void subscribe(int level, CallBack cb);

cb est une référence à une instance d'une classe implantée par le client permettant l'exécution d'une méthode coté client. L'interface *CallBack* fournit la méthode :

public void handle(int level);



Donnez l'implantation des programmes (interfaces et classes) pour *Client*, *CallBack* et *Server*.

QUESTIONS DE COURS

1) Expliquez la différence entre Web Services SOAP et Web Services REST.

2) Expliquez le problème qu'adressent les ESB et les principes de la solution qu'ils apportent.