TP3-5 : Java RMI

Bram De Wachter

February 28, 2005

1 Exécuteur de tâche

On désire continuer à utiliser des machines peu puissantes pour exécuter des applications dont certaines tâches peuvent être lourdes. Pour cela, une solution evisageable est d'utiliser des machines *hôtes* mettant à disposition leur puissance de calcul aux machines peu puissantes. La mise en place de ce genre de solution nécessite le déploiement de ce qu'on appelle des *exécuteurs de tâches*. Lorsqu'une application a un besoin important en puissance de calcul pour effectuer une tâche bien particulière, elle délaigue cette tâche à un exécuteur. Plusieurs exécuteurs de tâche peuvent être disponible. Pour être connu des applications clientes, ceux-ci ont plusieurs solutions. Une solution est qu'un distributeur de tâche maintient la liste de tous les exécuteurs, et que l'application en besoin de puissance de calcul soumet sa tâche au distributeur, qui le transmettra à l'exécuteur le moins occupé.

2 Exercice

On vous demande d'implémenter à l'aide de Java RMI

1. un exécuteur de tâche implémentant l'interface suivante :

```
package interfaces;
public interface Executor extends Remote {
  public Object execute(Task task) throws RemoteException;
  public int getLoad() throws RemoteException;
}
```

   Lorsque la méthode `execute` est appelée, l'exécuteur de tâche appellera la méthode `run` de la tâche. La méthode `getLoad` indiquera la charge de l'exécuteur.

2. un client envoyant une tâche aux exécuteurs. Chaque tâche implémentera l'interface suivante:

```
public interface Task extends java.io.Serializable {
    Object run();
}
```

   où la méthode `run` effectuera les calculs voulues

3. un distributeur de tâches, qui retient la liste des exécuteurs. C'est cette classe qui utilisera le `rmiregistry` pour s'annoncer aux clients et aux exécuteurs. Quand un exécuteur est lancé, il utilisera la méthode `register` pour se rajouter à la liste. La méthode `unregister` sera utilisé par les executeurs avant leur arrêt. La méthode `forwardTask` sera utilisé par les clients pour la distribution d'une tâche.

```
public interface Distributor extends Remote {
    public void register(Executor exec) throws RemoteException;
    public void unregister(Executor exec) throws RemoteException;
    public void forwardTask(Task t) throws RemoteException;
}
```

3 Plan de campagne

Pour se familiariser avec l'environnement RMI, commencez par implémenter l'exemple `Hello world!` qui se trouve dans le tutorial sur `http://java.sun.com/j2se/1.4.2/docs/guide/rmi/getstart.doc.html`, vous pourrez réutiliser pas mal de code dans cet exemple pour l'implémentation des exécuteurs.

Ensuite, changer dans l'example la méthode `sayHello()` pour en faire `execute(Task t)` et changez le client pour qu'il envoie une tâche de calcul directement à l'exécuteur (l'ancien `HelloImpl`).

Dans un troisième temps, implémentez la solution complète utilisant le distributeur de tâches. Question finale: comment la tâche peut-elle renvoyer des résultats intermédiaires vers le client depuis l'exécuteur, sans que ce dernier doit être modifié ? Implémentez cette solution.