**Programmation Objet Avancée** **Année universitaire 2019-2020**

# TP4 : RMI

Pour ce TP, nous allons développer des applications distribuées sur le réseau en utilisant RMI.

Les serveurs et les clients seront exécutés sur la machine sur laquelle vous travaillez.

## Exercice 1 :

Pour cet exercice, commencez par récupérer le dossier Hello.

1. Compiler et testez votre programme.
2. Préciser dans votre compte-rendu quelles sont les commandes à taper pour cela, ainsi que si elles doivent avoir lieu coté serveur ou cotée client.
3. Regarder et comprendre le code. Quel est le rôle de chaque fichier ?
4. Que se passe-t-il si plusieurs d'entre vous lancent le serveur sur la même machine ? Comment résoudre le problème ?
5. Quelle est la différence entre Naming.bind() et Naming.rebind() ? Quel problème peut-on rencontrer avec Naming.bind() ?
6. Est-il possible d'enregistrer un serveur sur un registry distant ? pourquoi ?

## Exercice 2 :

L'objectif de cet exercice est d'écrire un programme qui permet de calculer sur le serveur la fonction fibonnacci.

Vous pouvez exploiter l'interface ci-dessous :

```
package fibonaacci;
 import java.rmi.*;
public interface Serveur extends Remote
 {
 public int fibonnaci(int rang) throws RemoteException;
 }
```

1. Ecrire la classe ServeurImpl.java et Client.java qui contiennent respectivement l'implémentation du serveur et la demande du client (par exemple fibonnaci (5)).

## Exercice 3 :

Ecrire une application qui permet de mémoriser les résultats académiques d'un ensemble d'étudiants. L'application est constituée de deux interfaces : *Etudiant.java* et *Promotion.java*. L'interface *Etudiant.java* donne accès aux données associées à un étudiant : c'est à dire son nom, son prénom, son numéro d'étudiant ainsi qu'un ensemble de notes. Chaque étudiant passe plusieurs épreuves. Chaque épreuve donne lieu à une note. Chaque note est associée à un coefficient. Un coefficient est une donnée de type double dont la valeur est comprise entre 0 et 1. La somme des coefficients de toutes les épreuves d'un étudiant doit être égale à 1. Ces différentes épreuves sont stockées dans un ensemble d'instance de la classe Epreuve_avec_coeff.java. L'interface Etudiant.java propose trois méthodes :

- ajouter_une_epreuve: qui permet d'ajouter une épreuve à un étudiant. Une épreuve est constituée d'un nom (ex: "Ecrit de math"), d'une note et d'un coefficient indiquant le poids de l'épreuve dans la moyenne générale de l'étudiant.
- afficher_liste_des_epreuves: qui renvoie au client une chaine de caraactère contenant la liste des épreuves associées à l'étudiant. Cette chaine de caractères est donc construite par le serveur afin d'être affichée par le client à l'utilisateur
- calculer_la_moyenne: qui calcule la moyenne générale de toutes les épreuves d'un étudiant conformément aux coefficients de chaque épreuve.

L'interface Promotion.java permet:

- De créer un nouvel étudiant (méthodeajouter_un_etudiant).
- De rechercher un étudiant précédemment enregistré (méthoderechercher_un_etudiant). Grâce à la référence d'objet ainsi récupérée, le client peut alors demander le calcul de la moyenne générale de l'étudiant.
- De calculer la moyenne générale de toute la promotion (méthode calculer_moyenne_de_la_promotion).

1. Proposer un serveur ainsi qu'une classe d'implémentation pour chacune des interfaces accessibles à distance.
2. Proposer un client qui déclare plusieurs étudiants ayant passés plusieurs épreuves. Puis, tester votre solution en affichant les épreuves, la moyenne de chaque étudiant ainsi que la moyenne générale de la promotion.