Programmation Distribuée 2009
Premiers pas avec Corba

http://cafeteria.u-strasbg.fr/~gancars

Pour tous les exercices, vous ferez attention a toujours lancer les applications serveur et cliente sur deux machines DIFFERENTES.

Pensez bien à tuer tous vos processus à chaque fois que vous vous delogez d'une machine Linux.

Exercice 1

1. Rendez-vous dans le répertoire `Carre`
2. Réaliser la projection de l'interface IDL : `idlj -fall -oldImplBase Icarre.idl`
   Notez tous les fichiers `*.java` qui sont générés automatiquement. *IMPORTANT* : ouvrir les fichiers `Icarre.java` et `IcarreOperations.java`. En déduire la classe que vous devez utiliser pour réaliser l'implantation du service.
3. Ecrire la classe implantant le service.
4. Corriger le code du serveur (Serveur.java) et du client (Client.java)
5. Lancer la compilation complète par `UTILS/compile`'
6. Lancer l'application en commencant par le serveur par `java Serveur` qui normalement affiche son IOR (*Interoperable Object Reference*).
   (*ATTENTION : la chaine "IOR:" fait partie de l'IOR du serveur*)
7. Lancer le client : `java Client IOR_du_serveur nombre`
8. (a) Lancez le serveur sur une machine par `java Serveur`
   (b) Exécuter le client sur une autre machine par exemple : `java Client <IOR:...> 4`
   Question : Comment le client arrive-t-il à savoir sur quelle machine se trouve le serveur ? (java -jar UTILS/IORDecoder.jar <IOR>)

N'oublier pas de tuer le serveur en vous deconnectant !

Exercice 2

1. Recopier `Carre` dans un nouveau répertoire que vous appelerez `Carre_HOLDER`, (utiliser la commande 'cp -r')
2. Modifier la fonction `long carre(in long source)` afin de faire passer le resultat de la fonction en second paramètre. Modifier le fichier `Icarre.idl` en conséquence.
3. Regarder le fichier `IcarreOperations.java` et modifier le fichier `IcarreImpl.java` en conséquence.
4. Ecrire le code du serveur (Serveur.java) et du client (Client.java) et lancer l'application.

Attention Avant d'exécuter la commande "make", il faut déclarer le nom de l'interface par `export INTERFACE=<nom interface>`. Exemple : `export INTERFACE=Icarre`.

Exercice 3

Le fichier `OpMatrice.idl` définit l'interface IDL d'un service proposant l'addition de 2 matrices. En projetant l'interface *idl* grâce a l'outil `idlj`, regarder en quel type Java est projeté le type IDL `long`.

Proposer une implementation CORBA de ce service. Cette fois ci on pourra additioner des matrices rectangulaires de dimension $n \times k$.

Remarque : Pour éviter de faire un copier/coller de l'IOR à partir du shell, on peut écrire l'IOR dans un fichier avec le serveur puis le lire depuis le client. Dans le fichier IDL est défini une constante `iorfile` contenant le nom du fichier où l'on va stocker l'IOR.

Dans le serveur pour écrire l'IOR dans un fichier utilisez le code :

```
import java.io.*;
        ...
FileOutputStream file = new FileOutputStream(iorfile.value);
PrintWriter out = new PrintWriter(file);
out.println(ior);out.flush();
file.close();
```

Dans le client pour lire l'IOR dans un fichier utilisez le code :

```
import java.io.*;
    ...
FileReader file = new FileReader(iorfile.value);
BufferedReader in = new BufferedReader(file);
ior = in.readLine();
file.close();
```

Exercice 4

Modifier votre programme afin que le serveur et le client utilise un serveur de noms (tnameserv)

Exercice 5

1. Recopier `Carre` dans un nouveau répertoire que vous appelerez `Carre_POA`.
2. Utiliser le *Portable Object Adaptor* (POA) dans cette nouvelle version. Par rapport à l'exercice 1, un intermédiaire appelé `POA` se place entre le squelette de l'objet distant et l'ORB au niveau du serveur. Ce POA normalisé permet de faire interopérer le serveur avec differents ORBs (pas uniquement celui de java SUN) :
   - Vous modifierez le fichier `compile` en enlevant l'option `-oldImplbase` lorsque vous appelez `idlj`.
   - Vous verrez alors que la projection IDL vers java change alors (le squelette généré n'est plus `_IcarreImplBase` mais `IcarrePOA`)

Exercice 6

Le fichier `Annuaire.idl` définit l'interface IDL d'un service d'annuaire. Proposer une implementation CORBA de ce service équivalente à celle vue avec RMI. Pour cela, vous utiliserez le POA.