Middleware
D'autres petits pas avec Corba

Dans le répertoire `Nommage`, vous trouvez un exemple d'utilisation du service de noms. Cet exemple est compilable par `./compile` et exécutable (les commandes à exécuter sont présentées dans les fichiers *Serveur.java* et *Client*.java*).

Dans la suite, on vous demande de mettre au point une interface pour une banque. Vous utilisez uniquement le POA.

Exercice 1

1. Le fichier *Compte.idl* du répertoire `Banque(s)` décrit l'interface `Compte` dans laquelle sont définis les services suivants sur le compte d'un client :
   - dépose d'un certain montant sur le compte : `void deposeBillets(réel)`;
   - retrait d'argent du compte (mais vous n'autorisez pas le découvert !) : `booléen retireBillets(réel)`;
     Si le crédit sur le du compte est insuffisant, l'opération n'est pas realisée et `false` est retourné
   - affiche le montant en crédit sur le compte : `réel afficheMontant()`;
   - vire une somme du compte courant vers un autre compte spécifié en paramËtre :
     `booléen virementCompteaCompte(réel, Compte);`
     Si le crédit sur le du compte est insuffisant, l'opération n'est pas réalisée et `false` est retourné
2. Réaliser une implémentation *CompteImpl.java* pour cette interface *Compte.idl*.
3. Réaliser un serveur qui crée deux comptes client et les publie sur le serveur de noms. Les informations sur la machine hébergeant le serveur de noms seront éventuellement passées en argument du programme.
4. Réaliser un client qui :
   - ajoute une somme sur les deux comptes du Serveur,
   - retire une autre somme sur chaque compte du Serveur,
   - fait un virement d'un compte à l'autre.

   Entre chaque opération vous afficherez les montants restants sur les deux comptes.

Exercice 2

Construire un deuxieme serveur qui crée un contexte (dans le service de noms) correspondant à une agence bancaire d'une banque donnée. Dans cette agence bancaire, vous créerez deux contextes correspondants à deux noms de clients différents. Vous créerez enfin deux comptes par client qui seront associés à des objet CORBA de type `Compte`.

Exercice 3

Réaliser un client qui utilise le deuxiËme serveur et qui :

- ajoute une somme sur les 4 comptes du Serveur,
- retire une autre somme sur chaque compte du Serveur,
- fait un virement d'un compte a l'autre.

Entre chaque opération vous afficherez les montants sur les différents comptes

Exercice 4

1. Réaliser un client qui permet de lister les différentes agences puis les clients qui se trouvent dans ces agences (le nom de l'agence sera passé en argument du client). Vous utiliserez pour cela la methode `list` de la classe `NamingContext` (à vous de trouver l'API java).
2. Vérifiez vos solutions dans la configuration suivante :
   - le serveur de noms (tnameserv) sur une machine,
   - le serveur sur une deuxiËme machine différente de la premiËre,
   - le client sur une troisiËme machine différente des autres.

Exercice 5

On désire maintenant gérer plusieurs banques. Pour cela, on associe à chaque banque un serveur de noms différent. Par exemple, le serveur de noms de la banque CreditAlsacien sera sur le port 10023 de la machine A, alors que le serveur de noms de la banque DonneTesThunes sera sur le port 52410 de la machine B. Enfin, on trouvera sur la machine C, le serveur qui permet de retrouver les adresses des serveurs de noms associés aux différentes banques.

1. Décrire l'interface qui permet d'interroger le serveur afin d'obtenir l'adresse d'une banque ;
2. Écrire et lancer le serveur correspondant ;
3. Écrire et lancer un client qui vire une somme d'un compte d'un client d'une agence bancaire d'une banque vers le compte d'un client d'une agence bancaire d'une autre banque : ces informations sont données en paramËtres du client. En cas d'erreur (solde insuffisant ; client, agence ou banque inexistant ...), le message doit Ítre explicite