Exemple CORBA

Objectif :
Calculer un montant TTC à partir d'un montant HT et un taux de TVA en utilisant l'architecture CORBA.

Etape 1: Developper l'interface en IDL

```
module calcul
{
interface calcul_montants
{
double calcul_ttc(in double mt_ht, in double taux);
};
};
```

Etape 2 : Générer les fichier java client et serveur à partir de l'interface IDL :

```
D:\cnam\jdk\bin> idlj -td "D:\cnam\jdev\mywork\corbatest\corbatest"  -fall
"D:\cnam\jdev\mywork\corba\spec.idl"
```

Cette commande génère les fichiers suivants :

- _calcul_montantsStub : Stub de l'objet calcul_montants (côté client).
- calcul_montants : Interface Java de l'interface IDL spec.idl
- calcul_montantsOperations : Interface des méthodes de l'objet Calcul
- calcul_montantsPOA : Squelette de l'objet calcul_montants (côté serveur)
- calcul_montantsHelper : Le Helper de l'objet calcul_montants
- calcul_montantsHolder : Le Holder de calcul_montants

Etape 3 : Créer le servant (la classe qui fait le calcul) :

```
package calcul;
import org.omg.CosNaming.*;
//inclure le package des exceptions pouvant etre generees
// par le service de nommage
import org.omg.CosNaming.NamingContextPackage.*;
// sert a manipuler les objets CORBA
import org.omg.CORBA.*;
// Classes necessaires pour referencer le POA
import org.omg.PortableServer.*;
import org.omg.PortableServer.POA;
// Proprietes pour initialiser l'ORB
import java.util.Properties;
public class Calcul_Servant extends calcul_montantsPOA {
   public Calcul_Servant() {
   }
   public double calcul_ttc(double mt_ht, double taux) {
    return mt_ht*(1+taux/100);
```

```
    }
}
```

Etape 4 : Créer le serveur :

```
package calcul;

// le serveur va utiliser le service de nommage
import org.omg.CosNaming.*;
//inclure le package des exceptions pouvant etre generees
// par le service de nommage
import org.omg.CosNaming.NamingContextPackage.*;
// sert a manipuler les objets CORBA
import org.omg.CORBA.*;
// Classes necessaires pour referencer le POA
import org.omg.PortableServer.*;
import org.omg.PortableServer.POA;
// Proprietes pour initialiser l'ORB
import java.util.Properties;

public class Serveur {
    public Serveur() {
    }

    public static void main(String args[]) {
    try {
    // creer et initialiser l'ORB qui integre
    // le service de noms
    ORB orb=ORB.init(args, null);
    // obtenir la reference de rootpoa &
    // activer le POAManager
    POA rootpoa =
    POAHelper.narrow(orb.resolve_initial_references("RootPOA"));
    rootpoa.the_POAManager().activate();
    // creer le servant
    Calcul_Servant calc= new Calcul_Servant();

    // obtenir la reference CORBA du servant
    org.omg.CORBA.Object ref = rootpoa.servant_to_reference(calc);
    calcul_montants href = calcul_montantsHelper.narrow(ref);
    // obtenir la reference du contexte de nommage
    org.omg.CORBA.Object objRef =
    orb.resolve_initial_references("NameService");
    // Utiliser NamingContextExt qui est Interoperable
    NamingContextExt ncRef = NamingContextExtHelper.narrow(objRef);
    // enregistrer le servant dans le service de nommage
    String name = "calcul_ttc";
    NameComponent path[] = ncRef.to_name( name );
    ncRef.rebind(path, href);
```

```
   System.out.println("Server is waiting");
   // attendre les invocations des clients
   orb.run();

   } catch(Exception e) {
   System.err.println("Erreur : "+e);
   e.printStackTrace(System.out);
   }
   }


}
```

Etape 5 : Créer le client :

```
package calcul;

import org.omg.CosNaming.*; // inclure le service de nommage
import org.omg.CORBA.*; // manipuler des objets CORBA
import org.omg.CosNaming.NamingContextPackage.*;

public class Client {
   public Client() {
   }

   public static void main (String args[]) {
   try {
   double mt_ht;
   double taux;
   double mt_ttc;
   mt_ht = Double.valueOf(args[0]);
   taux = Double.valueOf(args[1]);

   // creer et initialiser l'ORB
   ORB orb = ORB.init(args, null);
   // obtenir une reference au service de nommage
   org.omg.CORBA.Object objRef =
   orb.resolve_initial_references("NameService");
   // Utiliser NamingContextExt au lieu de NamingContext.
   //car interoperable
   NamingContextExt ncRef =
   NamingContextExtHelper.narrow(objRef);

   // demander la reference de l'objet au service de noms
   String nom = "calcul_ttc";
   calcul_montants calcul_ttc = calcul_montantsHelper.narrow
   (ncRef.resolve_str(nom));
   // faire appel a l'objet serveur et imprimer les resultats
   mt_ttc = calcul_ttc.calcul_ttc(mt_ht,taux);
```

```
   System.out.println("le montant ttc "+ mt_ttc);
   }
   catch(Exception e) {
   System.out.println("Erreur : "+e);
   e.printStackTrace(System.out);
   }
   } // fin du main

}
```

Etape 6 : Demarrer l'ORB sur la machine serveur

```
start orbd -ORBInitialPort 1500
```

Etape 7 : Lancer le serveur :

```
java calcul.Serveur -ORBInitialPort 1500
```

Etape 8 : Executer le client :

```
java calcul.Client 100 20 –ORBInitialHost localhost –ORBInitialPort 1500
```