TDM04 d'Informatique Répartie
Corba en C et Java

ASI4 - INSA Rouen
CORRECTION

Calculatrice distribuée sur Complexes à l'aide de Corba, en Java et C

Le but de ce TP est de réaliser une calculatrice distribuée sur nombres complexes à l'aide de Corba. Pour cela, vous respecterez les contraintes suivantes :

- Votre calculatrice supportera 4 types d'opération : additions, soustractions, multiplications et divisions. Le serveur proposera donc 4 services distribués et ne travaillera que sur des complexes.
- Le serveur, effectuant le calcul, est en C et le client faisant les requêtes en Java. Vous utiliserez un service de nommage.
- Afin d'éviter de recoder les calculs de complexes, les fichiers Complexe.h et Complexe.c sont fournis. **Ils doivent être utilisés sans aucune modification.**
- Une exception sera levée en cas de division par 0.
- Vous réfléchirez bien sur le type d'éléments à transmettre (structures, objets ou objets non délocalisables).

NB : il s'agit d'Informatique Répartie donc pensez à tester votre programme avec serveur et client sur des machines différentes !

Correction

CalculatriceComplexes.idl

```
module CalculatriceComplexeCorba {
  struct Complexe {
    double re;
    double im;
  };

  interface Calculatrice{
    exception CalculatriceErreur {
      string raison;
    };
    Complexe additionner(in Complexe c1, in Complexe c2);
    Complexe soustraire(in Complexe c1, in Complexe c2);
    Complexe multiplier(in Complexe c1, in Complexe c2);
    Complexe diviser(in Complexe c1, in Complexe c2) raises (CalculatriceErreur);
  };

};
```

CompileIDL.sh

```
orbit-idl-2 --skeleton-impl --output-dir=Serveur CalculatriceComplexes.idl
idlj -fclient -keep -td Client CalculatriceComplexes.idl
```

CalculatriceComplexes-skelimpl.c

```
/* This is a template file generated by command */
/* orbit-idl-2 --skeleton-impl CalculatriceComplexes.idl */
/* User must edit this file, inserting servant */
/* specific code between markers. */

#include "CalculatriceComplexes.h"
#include "math/Complexe.h"

...
#if !defined(_impl_CalculatriceComplexeCorba_Calculatrice_additionner_)
#define _impl_CalculatriceComplexeCorba_Calculatrice_additionner_ 1
static CalculatriceComplexeCorba_Complexe
impl_CalculatriceComplexeCorba_Calculatrice_additionner(impl_POA_CalculatriceComplexeCorba_Calculatrice *servant,
const CalculatriceComplexeCorba_Complexe* c1,
const CalculatriceComplexeCorba_Complexe* c2,
CORBA_Environment *ev)
{
CalculatriceComplexeCorba_Complexe retval;
   Complexe z1 = complexe_creer(c1->re, c1->im), z2 = complexe_creer(c2->re, c2->im), z;
```

```
   z = complexe_additionner(z1, z2);
   retval.re = z.re;
   retval.im = z.im;
return retval;
}
#endif

#if !defined(_impl_CalculatriceComplexeCorba_Calculatrice_soustraire_)
#define _impl_CalculatriceComplexeCorba_Calculatrice_soustraire_ 1
static CalculatriceComplexeCorba_Complexe
impl_CalculatriceComplexeCorba_Calculatrice_soustraire(impl_POA_CalculatriceComplexeCorba_Calculatrice *servant,
const CalculatriceComplexeCorba_Complexe* c1,
const CalculatriceComplexeCorba_Complexe* c2,
CORBA_Environment *ev)
{
CalculatriceComplexeCorba_Complexe retval;
   Complexe z1 = complexe_creer(c1->re, c1->im), z2 = complexe_creer(c2->re, c2->im), z;
   z = complexe_soustraire(z1, z2);
   retval.re = z.re;
   retval.im = z.im;
return retval;
}
#endif

#if !defined(_impl_CalculatriceComplexeCorba_Calculatrice_multiplier_)
#define _impl_CalculatriceComplexeCorba_Calculatrice_multiplier_ 1
static CalculatriceComplexeCorba_Complexe
impl_CalculatriceComplexeCorba_Calculatrice_multiplier(impl_POA_CalculatriceComplexeCorba_Calculatrice *servant,
const CalculatriceComplexeCorba_Complexe* c1,
const CalculatriceComplexeCorba_Complexe* c2,
CORBA_Environment *ev)
{
CalculatriceComplexeCorba_Complexe retval;
   Complexe z1 = complexe_creer(c1->re, c1->im), z2 = complexe_creer(c2->re, c2->im), z;
   z = complexe_multiplier(z1, z2);
   retval.re = z.re;
   retval.im = z.im;
return retval;
}
#endif

#if !defined(_impl_CalculatriceComplexeCorba_Calculatrice_diviser_)
#define _impl_CalculatriceComplexeCorba_Calculatrice_diviser_ 1
static CalculatriceComplexeCorba_Complexe
impl_CalculatriceComplexeCorba_Calculatrice_diviser(impl_POA_CalculatriceComplexeCorba_Calculatrice *servant,
const CalculatriceComplexeCorba_Complexe* c1,
const CalculatriceComplexeCorba_Complexe* c2,
CORBA_Environment *ev)
{
CalculatriceComplexeCorba_Complexe retval;
   Complexe z1 = complexe_creer(c1->re, c1->im), z2 = complexe_creer(c2->re, c2->im), z;
   int erreur;
   CalculatriceComplexeCorba_Calculatrice_CalculatriceErreur* exception;
   erreur = complexe_diviser(z1, z2, &z);
   if (erreur==0) {
     retval.re = z.re;
     retval.im = z.im;
   }
   else {
     exception = CalculatriceComplexeCorba_Calculatrice_CalculatriceErreur__alloc();
     exception->raison = CORBA_string_dup ("Division par 0.");
     CORBA_exception_set(ev, CORBA_USER_EXCEPTION, ex_CalculatriceComplexeCorba_Calculatrice_CalculatriceErreur,
          exception);
   }
return retval;
}
#endif
```

serveur.c

```
#include <stdio.h>
#include <ORBitservices/CosNaming.h>
#include <ORBitservices/CosNaming_impl.h>
#include "CalculatriceComplexes.h"
#include "CalculatriceComplexes-skelimpl.c"

int main (int argc, char *argv[]) {

  PortableServer_POA rootpoa;
  CalculatriceComplexeCorba_Calculatrice corbaCalculatriceObject;
  CORBA_ORB orb=NULL;
  CORBA_Environment env;
  CosNaming_NamingContext nameServiceObject;
  CosNaming_NameComponent namePath[1] = {"CalculatriceComplexeCorba", ""};
  CosNaming_Name name = {1, 1, namePath, CORBA_FALSE};

  CORBA_exception_init(&env);
  orb = CORBA_ORB_init(&argc, argv, "orbit-local-orb", &env);

  rootpoa = (PortableServer_POA)CORBA_ORB_resolve_initial_references(orb, "RootPOA", &env);
```

```
  PortableServer_POAManager_activate(PortableServer_POA__get_the_POAManager(rootpoa, &env), &env);
  nameServiceObject = CORBA_ORB_resolve_initial_references(orb, "NameService", &env);

  corbaCalculatriceObject = impl_CalculatriceComplexeCorba_Calculatrice__create(rootpoa,&env);
  CosNaming_NamingContext_rebind(nameServiceObject, &name, corbaCalculatriceObject, &env);

  CORBA_ORB_run(orb, &env);

  CORBA_Object_release(corbaCalculatriceObject, &env);
  CORBA_ORB_shutdown(orb, CORBA_FALSE, &env);

  return 0;
}
```

compileServeur.sh

```
gcc -c math/Complexe.c

gcc -c CalculatriceComplexes-common.c `orbit2-config --cflags --use-service=name server`
gcc -c CalculatriceComplexes-stubs.c `orbit2-config --cflags --use-service=name server`
gcc -c CalculatriceComplexes-skels.c `orbit2-config --cflags --use-service=name server`
gcc -c CalculatriceComplexes-skelimpl.c `orbit2-config --cflags --use-service=name server`
gcc -c serveur.c `orbit2-config --cflags --use-service=name server`

gcc -o Serveur Complexe.o CalculatriceComplexes-common.o CalculatriceComplexes-stubs.o CalculatriceComplexes-
     skels.o CalculatriceComplexes-skelimpl.o serveur.o `orbit2-config --libs --use-service=name server`
```

CalculatriceSurComplexeCORBA.java

```
import java.io.*;
import java.util.*;
import java.net.*;
import org.omg.CORBA.*;
import org.omg.CosNaming.*;
import CalculatriceComplexeCorba.*;
import CalculatriceComplexeCorba.CalculatricePackage.*;

public class CalculatriceSurComplexeCORBA {

  CalculatriceComplexeCorba.Calculatrice calculatriceComplexeCORBA;

  public CalculatriceSurComplexeCORBA(String[] args) throws Exception {
    ORB orb = ORB.init(args, null);

    NamingContext corbaNamingServiceReference = NamingContextHelper.narrow(orb.resolve_initial_references("
        NameService"));
    NameComponent calculatriceName = new NameComponent("CalculatriceComplexeCorba", "");
    NameComponent calculatricePath[] = {calculatriceName};
    this.calculatriceComplexeCORBA = CalculatriceComplexeCorba.CalculatriceHelper.narrow(
        corbaNamingServiceReference.resolve(calculatricePath));
  }

  public Complexe additionner(Complexe c1, Complexe c2) throws Exception {
    return calculatriceComplexeCORBA.additionner(c1, c2);
  }

  public Complexe soustraire(Complexe c1, Complexe c2) throws Exception {
    return calculatriceComplexeCORBA.soustraire(c1, c2);
  }

  public Complexe multiplier(Complexe c1, Complexe c2) throws Exception {
    return calculatriceComplexeCORBA.multiplier(c1, c2);
  }

  public Complexe diviser(Complexe c1, Complexe c2) throws Exception {
    return calculatriceComplexeCORBA.diviser(c1, c2);
  }
}
```

Client.java

```
public class Client {
  public static void main(String[] args) throws Exception {
    CalculatriceSurComplexeCORBA calc = new CalculatriceSurComplexeCORBA(args[0].split(" "));
    CalculatriceComplexeCorba.Complexe un = new CalculatriceComplexeCorba.Complexe(1.0, 0.0),
                                       zero = new CalculatriceComplexeCorba.Complexe(0.0, 0.0),
                                       un_deux = new CalculatriceComplexeCorba.Complexe(1.0, 2.0);
    System.out.println("1 + (1+2i) = " + calc.additionner(un,un_deux));
    System.out.println("1 - (1+2i) = " + calc.soustraire(un,un_deux));
    System.out.println("1 x (1+2i) = " + calc.multiplier(un_deux,un_deux));
    System.out.println("(1+2i) / 1 = " + calc.diviser(un_deux,un));
    System.out.println("(1+2i) / (1+2i) = " + calc.diviser(un_deux,un_deux));
    System.out.println("1 / (1+2i) = " + calc.diviser(un,un_deux));
    System.out.println("(1+2i) / 0 = " + calc.diviser(un_deux,zero));
  }
}
```

nommage.sh

```
orbit-name-server-2 -ORBCorbaloc=1 -ORBIIOPIPv4=1 -ORBIIOPIPName=$1 -ORBIIOPIPSock=$2 --key=NamingService
```

Remarque : *Ne pas oublier l'option "-ORBIIOPIPv4=1" quand on lance le serveur en C...*

Remarques

— Vous aurez alors accès à la correction du TDM durant une semaine à compter de la fin du TDM.
— **Déposez un compte-rendu de 2 pages `TDMCorba2-NomPrenom.pdf` sur moodle chez TOUTES les personnes du binôme. Ce CR contiendra les informations que vous jugerez nécessaires.**
— Votre CR sera disponible pour vous lors de l'examen machine.