Intégration des Objets & des Services

CORBA : Common Object Request Broker Architecture

Permet de développer des applications distribuées selon une approche orientée objet. ~~indépendamment des plateformes matérielles et logicielles~~.
Il permet à des applications actives sur différentes machines d'un environnement réparti hétérogène de coopérer au travers d'objets répartis.

OMG : Object Management Group.

→ Etablir des spécifications pour l'intégration d'applications réparties hétérogènes.

OMA : Object Management Architecture.

deux Modèles existent

Modèle objet & Modèle de référence.
↓ CIS + Interfaces
↓ identifier & classifier les différents types d'objets constituant un sys réparti.

ORB : Object Request Broker.

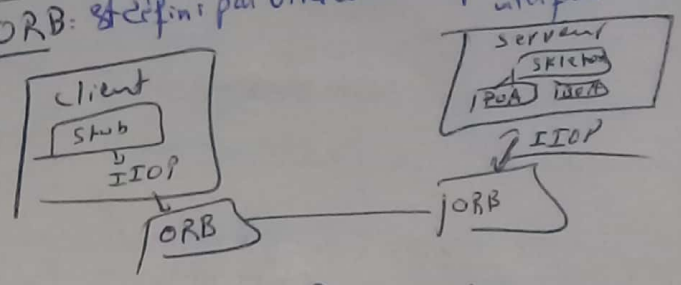
Permet de transporter les requêtes clients vers le serveur et les résultat vers le client en milieu hétérogène.

IDL : Interface Definition Language

IIOP : Internet Inter ORB Protocol.

GIOP

ORB : ~~est~~ défini par OMG comme un pseudo-objet avec interface.



IR : Interface Repository.

~~Stocke tout~~
Stocke toutes les définitions IDL des interfaces des objets du bus, des méthodes qu'ils supportent et des paramètres qu'ils nécessitent. → DIR

ImpIR : référentiel d'implantations → DSI.

Etapes de cycle de dév (78).

IOR : Interoperable Object Reference.

dépend du protocole (IIOP/GIOP).

Page (107)

Mise en œuvre d'un objet Corba.

↙ Par Héritage — Par délégation ↘

Page 109

Service de Nommage: (Page 124)

IDL :

* Langage de Spécification d'interfaces (orienté objet).
* fortement typé.
* indépendant de tout langage de programmation.
* langage de Spécification et non d'implantation.
* Il a été défini pour permettre la communication entre les applications clientes et les objets CORBA
* Utilisé pour générer des souches IDL coté client et des squelettes coté Serveur.

Eléments of IDL

Types; Constants; exceptions; modules; Interfaces
Operations; Attributs.

(146)

Constante

~~const~~ Const double PI = 3.14159;

Type

new type : typedef unsigned short Annee;

Type énumérés :

enum JourSemaine {Lundi, ---, Dimanche};

Struct :

Struct MaDate { Jour jour; Mois mois;
Annee annee; };

Union

Union DateMultiFormat switch (long)
{ case 1 : Date f1;
case 2 : String f1;
Default: Jour nbr;
};

Tableaux

Typedef long Vecteur [50];

Sequence

typedef sequence < float > Seq;

Module

module MyModule {
typedef long :d;
interface MyInterface {
=
};
};

Accès Externe d'un type T déclarer dans l'interface I du module M.

M::I::T

Opérations

long f (inout float c, in d, out b);

in : le Parametre est transmet vers l'objet
out : le Param est transmis par l'objet;
inout : echange dans les deux sens.

Attribute

attribute long p;
read only attribute long d;

⇔

long getP();
void setP(long pp);
long getd();

IDL Héritage

```
module m1{
exception exp1{string raison;};
interface Compte:C1, C2 {
void SayAny() raises (exp1);
};
```

Projection de IDL.

On peut avoir → squelettes
- le serveur en C++ : sockets.
- le client en java.

Environnement hétérogène...

mapping Java

```
module m1{
    interface I {
    attribute long C;
    void SayMsg(in string msg);
    };
};
```

client: org.omg.CORBA.Object
server: org.omg.PortableServer.Servant.

⇔

```
Public interface m1.I extends ... {
    Public int C();
    Public void C(int c);
    Public void sayMsg(java.lang.String msg);
    ...
};
```

Le mapping Java

Objectifs : utilisation d'objets CORBA depuis un programme Java.

Holder : pas de passage out & inout dans Java (passage par adresse).

La classe Holder Propose → solution pour gérer les paramètres de type **out** et **inout**

Helper : propose un ensemble des fonctions pour gérer les objet. Ex : → operation narrow pour remplacer les objets Corba en objet utilisateur.
* obtenir le repository ID.
* lecture et écriture dans un stream CORBA.

Exemple (182).

mapping des types

- void → void
- long → int
- long long → long
- char → char
- string → string
- float, double → float, double

module → Package.

Constant IDL → Constant Java

const long PI = 3,14; → public static final int PI = 3,14;

Structure → classe.

```
struct personne{string nom; string prenom;
→ public final class Personne {
    public String nom;
    public String prenom;
    public Personne();  public Personne(String ng, String pg)
    { nom = n; prenom = p; }
```

enum C{R,V,B}; → C val;
... (val.value) { case C.R: ... }

mapping Enum

IDL enum Mois {Jan, Fev, ---, Dec}

Java

```
Public final Class Mois {
Private int value;
Private Static int size = 12;
Private Static Mois[] array = new Mois[size];

Public Static final int _jan = 0;
 ''    ''    '' Mois Jan = new Mois(_Jan);
```

+ /// get value

```
Public Static Mois from_int (int value)
 } : if (value >= 0 && value < size)
        return array[value];
   else
     throw new org.omg.CORBA.BAD_PARAM();
 {
 Private Mois (int value)
       { m_value = val;
         array [val] = this; }
 };
```

Utilisation

```
Mois m = Mois.Janvier
Switch (m.value()) {
  case Mois._Jan : ——
    ——
{
m = Mois.from_int(7); if (m == Mois.Aout--);
```

Tableaux & Sequence IDL → Tableaux Java.

```
typedef long CodeB[3]
interface a {
 attribut codeB[3] cooki;
 }
```

⟺

```
public interface a {
 int[] code();
 void code(int[] newcode);
 {
```

POA

RootPOA

chaque POA peut AVoir des sous POA