Corba

Table des matières

Cette URL à l'air bien : http ://www.ciaranmchale.com/corba-explained-simply/

1 Taxonomie

- CORBA est une norme rédigée par l'OMG (Object Management Group).
- CORBA est un middleware 'objet'. Il repose sur la notion de bus logiciel, appelé ORB (Object Request Broker). Il existe plusieurs ORB.
- CORBA Les objets communiquent par l'intermédiaire du protocole IIOP (Internet Inter-ORB Protocol), basé sur TCP/IP. IIOP est la spécialisation d'un protocole plus général : GIOP (pour les autres réseaux). Depuis CORBA 2.0 tous les ORB sont compatibles via ce protocole.
- CORBA Les objets exportent leur définition dans ce qui est appellé une *interface* ; c'est la seule chose que connaisse les clients.
- CORBA Les *services* objets sont des routines génériques. Par exemple, le service *naming* permet de cataloguer les objets pour facilter leur connexion aux clients.
- IDL (Interface Definition Language) permet de définir les objets (méthodes, attributs exceptions).
- *L'adaptateur* d'objets assure leur gestion côté serveur. Il en existe 2 implémentations : le BOA (Basic Object Adapter) et le POA (Portable Object Adapter depuis CORBA 2.0).
- Le *servant* est l'implémentation de l'objet.
- Le *serveur* est l'application qui propose l'accès aux *servants*.
- la *référence d'objet* permettant de masquer le max de trucs côté *client*.
- Lors du BIND, côté client on récupère l'identifiant IOR de l'objet (Interoperable Object Reference).

2 Serveur de nom (service *naming*)

```
# apt-get install omniorb4-nameserver
```

Modifier le fichier */etc/omniORB4.cfg*

```
...
# Tracing level
#     level 10 - the above plus execution trace messages
traceLevel = 10
...
```

```
# InitRef
InitRef = bibiNameServiceRootContext=corbaname::localhost
InitRef = bibiServiceName1=corbaname::localhost
...

# /etc/init.d/omniorb4-nameserver restart
# tail -f /var/log/omniorb4-nameserver.log
```

3 Exemple en C++

Exemple copié/collé depuis cette url.

- Interface IDL :
  - Data.idl
- Server :
  - CORBA *server*
    Server.cpp
    This will remain the same for most CORBA servers. The only difference will be the name of the services (bibi...) and the implementation of the server functions/methods (CServiceA_i()).
  - *servant* implémentation : (cf "_i") This is the routine which performs the duties of the remote service on the server.
    CServiceA.h
    CServiceA.cpp
  - MakeServer
- Client :
  - *référence d'objet* The constructor is fairly generic and will be edited to match the names of your name server and service names. The constructor establishes the connection with the CORBA server but does not invoke any of the remote procedures. The routines "A" and "B" are called locally but executed remotely on the CORBA server. The CORBA calls CallServiceRoutineA() and CallServiceRoutineB() will be performed on the CORBA server. The interface between client and server for these functions are defined in the IDL.
    CRequestServiceA.h
    CRequestServiceA.cpp
  - CORBA *client* The constructor establishes the link with the CORBA server. The subsequent calls to Routine A and B are processed remotely on the CORBA server but called like any other local function call.
    Client.cpp
  - MakeClient

```
# apt-get install g++ omniorb4 omniorb4-doc

$ make -f MakeServer Server
$ ./Server
                                $ make -f MakeClient Client
                                $ ./Client
```

4 Exemple en python

Exemple copié/collé depuis cette url

```
# apt-get install omniidl4-python python-omniorb
```

echo.py

```
$ echo.py server omniORB
```

```
$ echo.py client omniORB
```

Dans le cas d'OMNIORB on peut simplifier comme ceci :

- testPython.idl
- server.py
- client.py