République **A**lgérienne **D**émocratique et **P**opulaire
Ministère de **l**'Enseignement **S**upérieur et de la **R**echerche **S**cientifique

UNIVERSITÉ **M**OULOUD **M**AMMERI DE **T**IZI-OUZOU

FACULTE DE GENIE ÉLECTRIQUE ET DE L'INFORMATIQUE

Module :Architechture Logiciel

Présentation de la technologie CORBA

Spécialité : INGÉNIERIE DES SYSTÈMES D'INFORMATION

NOM ET PRENOM :

AIT-IKENE Nadjib	*Gr1*
BOUCHERK Salim	*Gr1*
BOUSSAKOU Noureddine	*Gr1*
KACETE Djouher	*Gr2*
HADJ NACEUR Rayane	*Gr1*
TIGHREMT Zineb	*Gr2*

Pr : M.KERBICHE

Année Universitaire : 2022/2023

Table des matières

1 L'Architechture CORBA

1.1 Qu'est ce que le cadre CORBA ?

CORBA est l'acronyme de ***C**ommon **O**bject **R**equest **B**roker **A**rchitecture*. Il s'agit d'un standard développé en 1992 par Object Management Group (ou OMG) ; un consortium de plusieurs centaines d'entreprises dédiées à la construction de matériel informatique et l'édition de logiciels.
L'objectif est de développer des applications distribuées indépendantes de la plate-forme et du langage à travers des principes de conception orientés objet.

Corba repose sur la notion de bus logiciel (Object Request Broker). En Corba, tout est objet. On bénéficie donc des mécanismes d'héritage, polymorphisme et encapsulation.

1.2 Les avantages de CORBA

- ✓ **Interopérabilité :** Les objets Corba communiquent par l'intermédiaire du protocole IIOP (Internet Inter-ORB Protocol). Ce protocole permet la communication entre entités quelconques supportant le protocole TCP/IP.
- ✓ **Intégration aux systèmes existants :** Tout code existant peut être encapsulé dans un objet Corba (wrapping). Des passerelles existent pour des standards de distribution d'objets du marché (DCOM, DCE).
- ✓ **flexibilité du développement :** Les services rendus par les objets sont définis par leur interface qui tient lieu de contrat entre l'utilisateur et le fournisseur du service. Les parties définition et implantation d'un objet sont totalement dissociées.

1.3 Objets distribués de CORBA

- Localisation indifférente dans le réseau ;
- Sur des systèmes hétérogènes ;
- Implantés dans des langages différents.

1.4 Composants logiciels

- Accédés par n'importe quel client ;
- Via des invocations de méthodes ;
- Le client ne connait que l'interface publiée par le serveur ;
- L'interface est spécifiée dans un langage standard l'IDL.

1.5 Le modèle objet client-serveur CORBA

Il est fondé sur une entité virtuelle, l'objet CORBA, gérée par le bus CORBA. Chaque application peut exporter ses services sous la forme d'objets CORBA. La coopération client-serveur se déroule de la manière suivante :

1. Le client détient une référence sur un objet CORBA qui permet de le localiser sur le bus.

2. Le client dispose de l'interface de l'objet CORBA (type abstrait de l'objet CORBA) qui définit ses opérations et ses attributs (exprimés dans le langage IDL).
3. Le client réalise une requête (invoque) une opération sur l'objet CORBA.
4. Le bus CORBA achemine cette requête vers l'objet CORBA tout en masquant les problèmes d'hétérogénéité liés aux langages, systèmes d'exploitation, machines, réseaux.
5. L'objet CORBA est associé à un objet d'implantation
6. Le serveur détient l'objet d'implantation qui code l'objet CORBA (cette implantation pouvant évoluer au cours du temps) et gère son état temporaire.

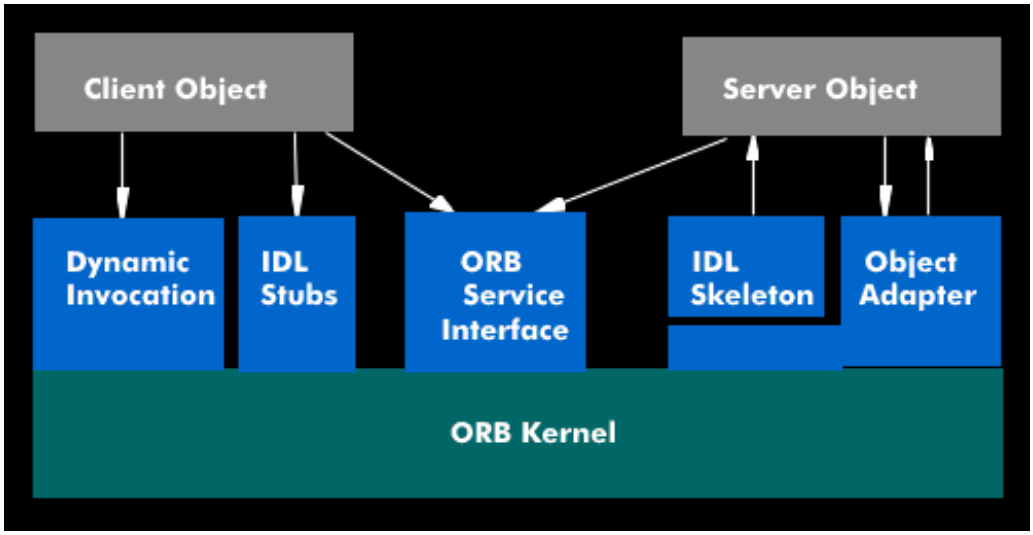


FIGURE 1 – CORBA Architechtur

1.6 Architecture CORBA

La relation entre objets distribués est une relation client-serveur. Le serveur fournit une interface distante. Le client invoque une interface distante. Le client peut lui-même jouer le rôle de serveur s'il implante une interface qui peut être invoquée à distance par d'autres objets.

Coté client

Le client connaît l'interface d'un objet spécifique (il dispose de l'IDL). Le client possède une représentation de l'objet distant appelée talon ("stub") générée par l'IDL. Le talon transmet la requête à l'objet distant via le bus logiciel (ORB). Il s'agit d'une représentation de l'objet distant responsable de la transmission d'une invocation de méthode du talon vers l'objet distant. Il ne s'agit pas d'une copie de l'objet distant. A travers le talon le client voit l'interface de l'objet sans savoir où est l'objet et sans connaître le langage de programmation utilisé pour son implantation.

En pratique, le client :

1. créé une instance locale d'ORB
2. récupère la référence du serveur de noms
3. récupère la référence de l'objet
4. invoque la méthode sur l'objet distant

Coté serveur

L'ORB utilise un squelette de code pour traduire l'invocation distante en un appel de méthode sur l'objet local. Le squelette traduit l'appel et les paramètres dans le format de l'implantation spécifique et appelle la méthode invoquée. Au retour de la méthode, les résultats (ou erreurs) sont traduits par le squelette et renvoyés au client via l'ORB. Communication entre ORB, protocoles réseaux Tout bus à la norme CORBA 2.0 doit fournir les protocoles GIOP et IIOP.

GIOP est un protocole générique. Il fournit :

- une représentation commune des données : CDR (Common Data Representation)
- un format de référence d'objet interopérable : IOR (Interoperable Object Reference)
- un ensemble de messages de transport de requêtes aux objets (request, reply, ...)

Les ORBs partagent un protocole commun IIOP ((Internet Inter ORB Protocol), implantation de GIOP sur TCP/IP et donc de l'internet. Ce protocole définit comment les ORBs (répondant à la spécification CORBA) se transmettent de l'information.

Autres services fournis par les ORBs :

- Recherche d'objets par leur nom
- Maintenance d'objets persistants
- Support pour le traitement transactionnel ...

2 IDL

2.1 Définition du langage IDL :

IDL ou **L**angage de **D**onnées **I**nteractif est un langage de programmation utilisé pour créer des applications effectuant une analyse de données, il est principalement utilisé par les astronomes et les experts en imagerie médicale ou le traitement d'image numérique nécessite une vitesse élevée dans de nombreuses applications.
Il est issu du logiciel interactif de traitement de données solaires ANA conçu par le groupe de physique solaire de la NASA au début des années 1980 pour le traitement des données du satellite Solar Maximum Mission. Aujourd'hui IDL s'est imposé comme LE système de traitement universel utilisé en physique solaire, et son succés débord désormais vers tous les horizons astronomiques.

2.2 une procédure IDL :

Un programme IDL se rédige à l'intérieur d'un éditeur de texte indépendant selon :
Pro nom_programme
Suite d'Instructions
end
Si ce fichier programme porte le nom nom_programme.pro on le compile en tapant sur le prompt IDL la commande suivante :
IDL> .run nom_programme
puis on l'exécute en appelant le programme directement par son nom :
IDL> nom_programme
On peut passer des variables à une procédure en entrée sortie. Par exemple avec var_in en entrée et var_out en sortie :
Pro nom_programme, var_in, var_out
Suite d'Instructions
End
puis on exécute la procédure en appelant le programme directement par son nom :
IDL> nom_programme, var_in, var_out

2.3 Structure syntaxique d'un module IDL :

<module> /* un contexte ou espace de nommage */

<déclaration de types>
<déclaration de constantes>
<déclaration d'exceptions>

interface /* une classe */
<déclaration de types>
<déclaration de constantes>
<déclaration d'exceptions>

<déclaration d'attributs> /* variables */
<déclaration d'opérations> /* méthodes */

<modules>

2.4 Les variables dans IDL :

IDL travaille avec des variables de tout type :

- Octets 8 bits de 0 à 255 (type BYTE)
- Entiers courts 16 bits signés de –32768 à 32767 (type FIX ou INT)
- Entiers courts 16 bits non signés de 0 à 65535 (type UINT)
- Entiers longs 32 bits signés (type LONG)
- Entiers longs 32 bits non signés (type ULONG)
- Réelles 32 bits dits simple précision (type FLOAT), 7 chiffres significatifs entre ±1038
- Réelles 64 bits dites en double précision (type DOUBLE), 14 chiffres significatifs
- Complexes 64 bits (type COMPLEX), 7 chiffres significatifs entre ±1038
- Complexes 128 bits (type DCOMPLEX), 14 chiffres significatifs entre ±10308
- Chaînes de caractères (type STRING)

2.5 Le contrat IDL :

Il permet d'exprimer, sous la forme d'un contrat, la coopération entre les fournisseurs et les utilisateurs de services. Il sépare l'interface des objets de leur implantation et masque les problèmes liés à la localisation des objets, à l'interopérabilité et à l'hétérogénéité. Il spécifie sous la forme d'interfaces IDL les types manipulés par un ensemble d'applications réparties.
Le contrat IDL rend transparent aux fournisseurs et clients l'infrastructure logicielle et matérielle. Il met client et fournisseur en relation à travers un bus CORBA.
La compilation d'un contrat IDL créé une souche (talon) IDL (ou SII, interface d'invocation statique) dans l'environnement de programmation du client et une souche (squelette) IDL (ou SSI, interface de squelette statique) dans l'environnement de programmation du fournisseur. Le client invoque localement la souche pour accéder à l'objet. La souche construit alors la requête qui est ensuite transportée par le bus logiciel pour être délivrée au squelette IDL (coté serveur donc) qui la délègue à l'objet.

2.6 Mappings :

Un mapping est une traduction des éléments fournis par l'IDL en éléments d'un langage de programmation.
Les principaux mappings disponibles sont : C, C++, Java, Smalltalk, CLOS, Python, Ada95, Cobol Objet.
Les compilateurs IDL ne traduisent que des squelettes. Les objets clients utilisent ces squelettes pour déterminer les opérations légales qu'ils peuvent invoquer sur un serveur. Les objets serveur fournissent une implantation pour ces squelettes.

2.7 Référentiel d'interfaces :

C'est une base de données en ligne de définitions d'objets, d'interfaces, de modules. Les définitions d'objets sont fournies par les compilateurs d'IDL ou par des fonctions d'écriture du référentiel. Les référentiels d'interfaces peuvent se fédérer et coopérer à travers les ORBs.

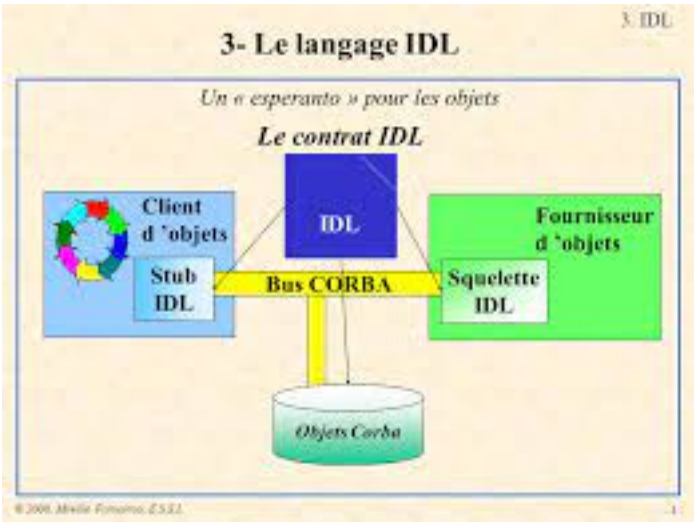


Figure 2 – IDL

3 Service de Nommage

3.1 Besoin d'un serivce de nommage

1. Initialiser le bus CORBA : obtenir l'ORB
2. Initialiser l'adaptateur d'objets : obtenir le POA
3. Créer les implantations d'objets
4. Enregistrer les implantations par l'adaptateur
5. **Diffuser leurs références (IOR)**
   - afficher une chaîne codifiant l'IOR ou
   - stocker l'IOR dans un fichier

   OU Utiliser un service de nommage
6. Attendre des requêtes venant du bus
7. Destruction du Bus

Besoins de nommage et client CORBA

1. Initialiser le bus (ORB)
2. Créer les souches des objets à utiliser
   (a) obtenir les références d'objet (IOR)
   - copier/coller l'IOR affichée coté serveur ou
   - lire le fichier contenant l'IOR
     OU Accéder au service de nommage

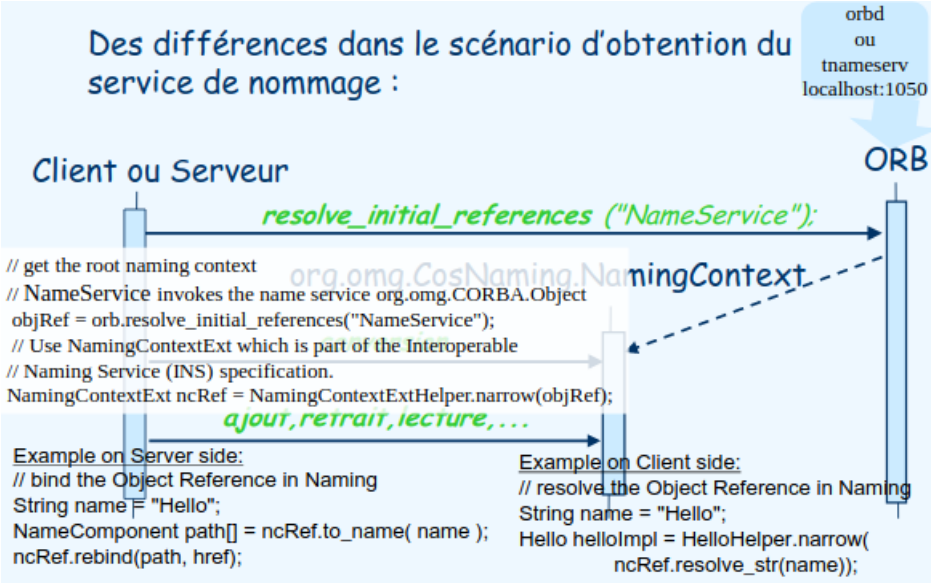
   (b) convertir vers les types nécessaires (narrow)

3.2 Les services de nommage : usage

Les services de nommage (ex : rmiregistry) sont utilisés :

- Pour stocker des objets
- Pour offrir un point d'accès aux applications réparties
- Référentiels d'entreprise pour accéder à :
  - ➦ des applications (machine/port),
  - ➦ des bases de données,
  - ➦ des informations de sécurité (gestion des accès au sein d'une entreprise)
  - ➦ des dispositifs tels que les imprimantes

3.3 Service de nommage pour CORBA :



4 Interopérabilité avec CORBA

Objectif = étendre l'interopérabilité entre des ORB différents,voir des systèmes objets "non CORBA".

Que faut-il pour interopérer ?

1. Manipuler un modèle objet identique.
2. tre capable d'interpréter convenablement une référence d'objet (ex : localiser son implémentation).
3. Être capable de véhiculer les invocations et les exceptions.
4. Manipuler une représentation communes des types, indépendamment des systèmes sous jacent (condition nécessaire dans un environnement constitué d'un seul ORB).

☞ Solution apportée dans CORBA 2 (CORBA 1 normalise uniquement l'interface de programmation). La version 2 définit :

— Des mécanismes de ponts.

— Un protocole standard : [**GIOP**] , ainsi que sa correspondance sur TCP/IP : IIOP

— Des environnements spécifiques (ex : interopérabilité avec DCOM ou par DCE).

☛ Compatible CORBA 2 = API CORBA + [**IIOP**].

GIOP : Global Inter-ORB Protocol.
IIOP : Internet Inter-ORB Protocol

Utilisation native de IIOP.

- Interopérabilité d'office.
- Le plus courant dans les ORB commercialisés aujourd'hui.
- Permet la mise en œuvre de protocoles optimisés, adaptés à certains types d'environnements ou d'applications.
- Conservation de l'existant.
- Le pont mémorise des objets proxies
- Conversion de protocole, de référence d'objet, voir de modèle objet.
- Combinaison de ponts + IIOP en fond de panier.

5 CORBA vs REST

REST n'est pas basé sur les protocoles mais c'est bien un ensemble de principes architecturaux. C'est une architecture communication point à un point, non fiable contrairement à CORBA. Il ne nécessite pas un outil standard mais un URI standard pour l'échange de formation. Il prend en charge JSON, XML et les microformats.La charge peut etre de n'importe quel format tout comme CORBA. Contient un gestionnaire d'erreur intégré.

	SOAP	REST
Architecture	SOAP is basically a SOA based architecture that can be used for middleware interoperability.	REST is basically an architecture designed for web based communication assuming only point to point communication
Reliability	Reliable	Not Reliable
Requirements	Requires standard tools with middleware support	Doesn't require any standard tools but requires a standard URI for information exchange
Communication Language	Works purely on XML based on WSDL standards	Supports JSON, XML, Microformats etc.
Payload	Payload must support SOAP schema	Payload can be of any format
Error Handling	Doesn't support error handling	Has built-in error handling

6 CORBA vs SOAP

La technologie middleware CORBA est plus rapide que le protocole de communication standard SOAP lors des échanges, et ce à cause du nombre d'informations qu'impose le format XML sur lequel est bâti SOAP,

Par contre lorsque le volume de données transmis par SOAP est faible par rapport au volume total de données échangés ce n'est plus un handicap.

Références

[1] http://deptinfo.cnam.fr/Enseignement/CycleProbatoire/CDI/ProjetCDI/HelloWorld/index.htm

[2] https://datascientest.com/definition-et-avantages-de-linfrastructure-corba

[3] https://lesia.obspm.fr/perso/jean-marie-malherbe/cours/Intro-IDL.pdf

[4] https://studylibfr.com/doc/2453396/les-services-de-nommage

[5] http://beru.univ-brest.fr/~singhoff/ENS/UE_objets_repartis/CM/corba1.pdf

[6] https://blog.udemy.com/wp-content/uploads/2014/05/S15.png

** FIN **