Présentation de l’architecture CORBA

Common Object Request Broker Architecture

Yves LALOUM

Conseil Audit de Systèmes d ’information CISA

ylaloum@advisehr.com

1.Introduction

· Depuis 1989, une association internationale l ’OMG (Object Management Group) définit la spécification de l'architecture d'un système à objets répartis, appelée CORBA (Common Object Request Broker Architecture /Architecture d’arbitrage de demande d'objets commun).

· CORBA est né du besoin de faire communiquer ensemble des applications en environnement hétérogène (plusieurs systèmes et plusieurs langages).

CORBA associe les concepts d'objet et de répartition dans une approche à la fois intégrée et appliquée.

Introduction

- L'intégration se fait à travers la notion d'invocation à distance d'objet (l'objet est l'unité de répartition), et l'application se fait à travers les services de répartition qui sont organisés sous forme de bibliothèques de classes.
- Dates importantes :
  - 1991 CORBA 1.1 Définition de l'Interface Definition Language (IDL)
  - et Application Programming Interfaces (API) interaction avec l'implémentation de l' Object Request Broker (ORB).
  - 1995 CORBA 2.0 Interopérabilité entre ORB et le protocole IIOP

2. L ’OMG (Object Management Group)

- Un consortium international à but non lucratif fondé en Avril 1989
- Plus de 900 membres (éditeurs de logiciels, constructeurs, utilisateurs)
- Basé aux Etats-Unis avec des représentants dans le monde entier
- Est financé à 95 % par ses membres et à 5% par l ’édition d’ouvrages
- Un site : HTTP://www.omg.org

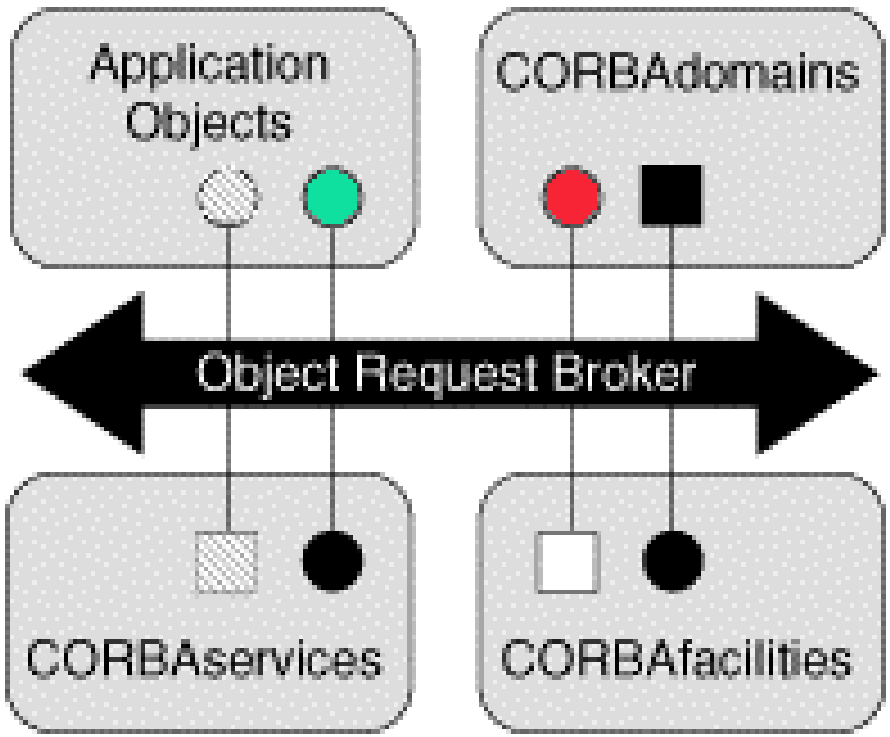
OMG: Les objectifs

- Fixer une définition du paradigme « objet »
- vise à promouvoir les technologies orientées objets pour la conception d'applications informatiques distribuées, interopérables et ouvertes.
- Spécifier seulement les interfaces : laisser la concurrence ouverte pour les implémentations
- Supporter de multiples langages (pas seulement Java et C++)
- Promouvoir ses choix technologiques (publications, forums, conférences, séminaires,formation….)
- Définir l'Object Management Architecture Guide (OMA) son modèle d'objets distribués et interopérables.

OMA Object Management Architecture

- Il définit l'architecture de gestion des objets et contient le fondement des standards :
  - Le modèle objet abstrait
  - le modèle de référence
  - Un glossaire des termes utilisés
- Il regroupe essentiellement 4 composants :
  - Le modèle objet
  - Corba Facilities
  - Corba Domains
  - Corba Services

OMA Object Management Architecture Les composants



- **CORBA Services** : Services d'objet commun
- **CORBA Facilities** : Utilitaires Communs
- **CORBA Domains** : Interfaces de domaine.
- **Application Objects** : Objets applicatifs

CORBA Services :

- les services objet communs (Common Object Services) proposent des services horizontaux pour faciliter l'utilisation courante des objets CORBA.
- **Naming Service** : Le service Nommage est l'équivalent des « pages blanches »
    - les objets sont désignés par des noms symboliques.
    - Cet annuaire est matérialisé par un graphe de répertoires de désignation.
- **Trader Service** : Le service Vendeur est l'équivalent des « pages jaunes » les objets peuvent être recherchés en fonction de leurs caractéristiques.

CORBA Services

- **Life Cycle Service :** Le service Cycle de Vie décrit des interfaces pour la création, la copie, le déplacement et la destruction des objets sur le bus. Il définit pour cela la notion de fabriques d'objets («Object Factory»).
- **Properties Service :** Le service Propriétés permet aux utilisateurs d'associer dynamiquement des valeurs nommées à des objets. Ces propriétés ne modifient pas l'interface IDL, mais représentent des besoins spécifiques du client comme par exemple des annotations.
- **Relationship Service :** Le service Relations sert à gérer des associations dynamiques (appartenance, inclusion, référence, auteur, emploi,...) reliant des objets sur le bus. Il permet aussi de manipuler des graphes d'objets.

CORBA Services

- **Externalization Service** : Le service Externalisation apporte un mécanisme standard pour fixer ou extraire des objets du bus. La migration, le passage par valeur, et la sauvegarde des objets doivent reposer sur ce service.
- **Persistent Object Service :** Le service Persistance offre des interfaces communes à un mécanisme permettant de stocker des objets sur un support persistant. Un objet persistant doit hériter de l'interface «Persistent Object» et d'un mécanisme d'externalisation.
- **Object Query Service** : Le service Interrogations permet d'interroger les attributs des objets. Il repose sur les langages standards d'interrogation comme SQL3 ou OQL.

Corba Services

- **Collection Service** : Le service Collections permet de manipuler d'une manière uniforme des objets sous la forme de collections et d'itérateurs.
- **Versionning Service** : Le service Changements permet de gérer et de suivre l'évolution des différentes versions des objets. Ce service maintient des informations sur les évolutions des interfaces et des implantations.
- **Security Service** : Le service Sécurité permet d'identifier et d'authentifier les clients, de chiffrer et de certifier les communications et de contrôler les autorisations d'accès. Ce service utilise les notions de serveurs d'authentification, de clients/rôles/droits (et délégation de droits), d'IIOP sécurisé (utilisant Kerberos ou SSL).

CORBA Services

- **Object Transaction Service** : Le service Transactions assure l'exécution de traitements transactionnels impliquant des objets distribués et des bases de données en fournissant les propriétés Atomicité, Cohérence, Isolation, Durabilité.
- **Concurrency Service** : Le service Concurrence fournit les mécanismes pour contrôler et ordonnancer les invocations concurrentes sur les objets. Le mécanisme proposé est le verrou. Ce service est conçu pour être utilisé conjointement avec le service Transactions.
- **Event Service** : Le service Evénements permet aux objets de produire des événements asynchrones à destination d'objets consommateurs à travers des canaux d'événements.

CORBA Services

- **Notification Service** : Dans Le service Notification les consommateurs sont uniquement notifiés des événements les intéressant.
- **CORBA Messaging [OMG-CM]** :Le service Messagerie définit un nouveau modèle de communication asynchrone permettant de gérer des requêtes persistantes lorsque l'objet appelant et l'objet appelé ne sont pas présents simultanément sur le bus.
- **Time Service** : Le service Temps fournit des interfaces permettant d'obtenir une horloge globale sur le bus (Universal Time Object), de mesurer le temps et de synchroniser les objets.
- .

CORBA Services

- **Licensing Service** : Le service Licences permet de mesurer et de contrôler l'utilisation des objets, et cela en vue de facturer les clients et de rémunérer les fournisseurs

CORBA Facilities

- les utilitaires communs définissent un cadre de haut niveau pour la création de logiciels à l'aide de composants réutilisables.
- **la gestion des tâches**
  - flux d ’activités (workflow)
  - gestions des transactions
- **l'interface utilisateur**
  - gestion du rendu . Gestion du bureau
  - affichage & dialogue utilisateur - présentation de l ’environnement gestion des documents composites .
  - Scripts support de l ’utilisateur

CORBA Facilities

- **la gestion de l'information**
  - modélisation . échange
  - règles de structuration, manipulation - GIF, JPEG, HTML,
  - stockage structuré . Codage et représentation
  - archivage de l ’information - physique de l ’information
- **l'administration système**
  - instrumentation . Ordonnancement
  - collecte d ’informations - traitement répétitif + événements
  - collecte des données . sécurité
  - journal d ’utilisation - politique de sécurité
  - qualité du service . gestions des événements
  - suivi d ’instance

Les interfaces de domaines CORBA

- les interfaces de domaines concernent des marchés verticaux
- ils définissent des interfaces spécialisées répondant aux besoins spécifiques d'un marché . **Il sont déterminés** au sein de groupes de travail nommés selon les cas :
  - « Working Groups »
  - « Domain Task Forces »
  - « Special Interest Groups »
  - …..

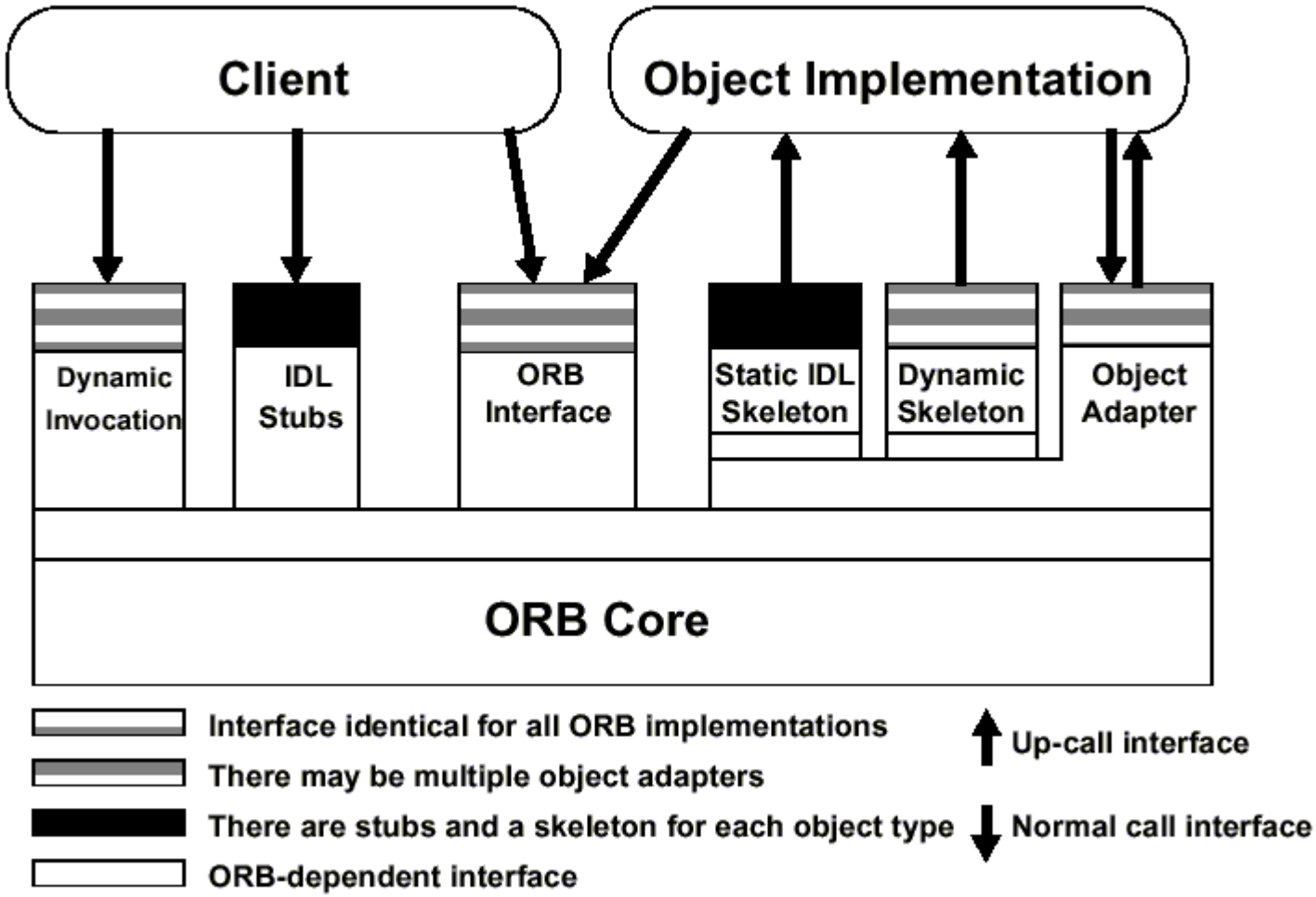
Les interfaces de domaines CORBA

- Quelques exemples :
  - Business Objects DTF pour la conception d'applications métier au dessus de CORBA.
  - CORBAmed définit les standards OMG pour le domaine de la médecine. Ces standards permettront l'interopérabilité entre les acteurs de la santé en définissant par exemple la notion d'objets patient.
  - Telecom DTF fait la liaison entre le monde CORBA et le monde des télécommunications.
  - Internet Platform SIG travaille sur les rapprochements des technologies CORBA avec les technologies d'Internet. Il étudie principalement les relations avec le World Wide Web.

OBJETS Applicatifs :

- Les objets applicatifs sont ceux qui sont spécifiques à une application répartie et ne sont donc pas standardisés.
- Toutefois, dès que le rôle de ces objets apparaît dans plus d'une application ils peuvent alors rentrer dans une des catégories précédentes et donc être standardisés par l'OMG.

BUS D'OBJETS DISTRIBUES



BUS D'OBJETS DISTRIBUES

- C'est le cadre visible que doivent respecter les développeurs de bus CORBA.
- Il correspond à l'approche intégrée du modèle
- Il assure le transport des requêtes entre tous les objets CORBA.
- Il offre un environnement d'exécution aux objets masquant l'hétérogénéité liée
    - Aux langages de programmation,
    - Aux systèmes d'exploitation,
    - Aux processeurs et aux réseaux.

BUS D'OBJETS DISTRIBUES

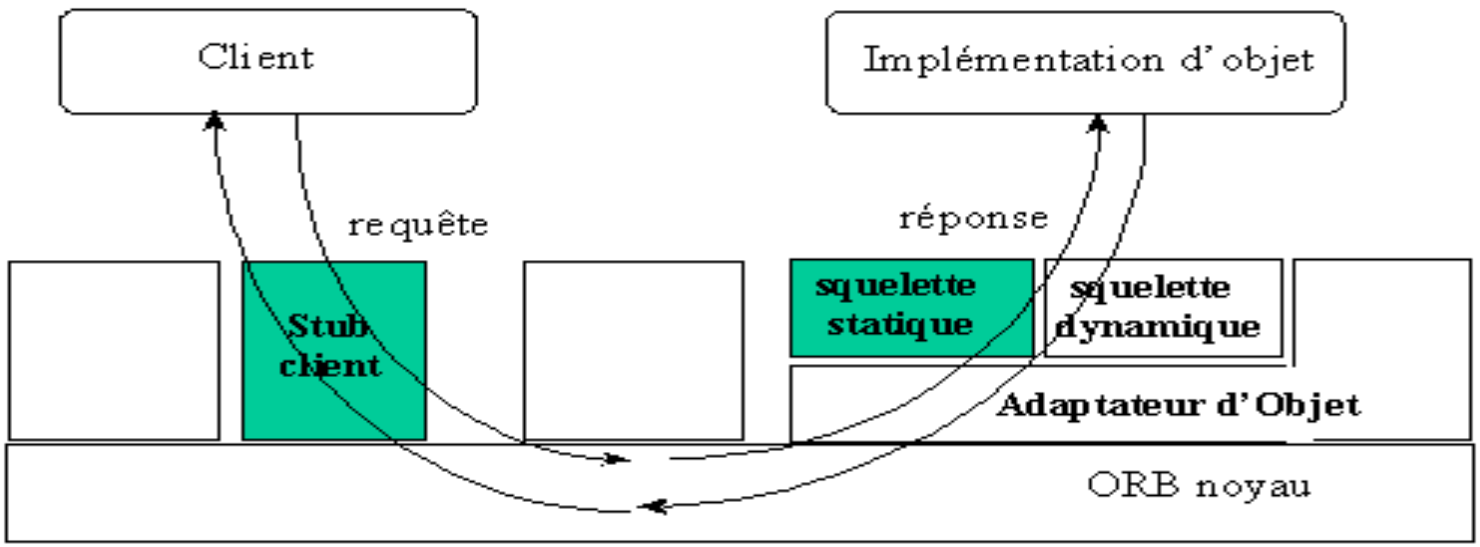
- Le bus CORBA est donc l'intermédiaire/négociateur à travers lequel les objets vont pouvoir dialoguer.
- Il fournit les caractéristiques suivantes :
  - La liaison avec « tous » les langages de programmation : cependant, actuellement l'OMG a seulement défini officiellement cette liaison pour les langages C, C++, SmallTalk, Ada, COBOL et Java
  - **La** transparence des invocations : les requêtes aux objets semblent toujours être locales, le bus CORBA se chargeant de les acheminer en utilisant le canal de communication le plus approprié.

BUS D'OBJETS DISTRIBUES

- **Niveau du processus :** les objets sont dans le même espace mémoire que les clients. C'est le cas typique des applications embarquées.
  - Ici, le langage OMG-IDL sert à spécifier les objets.
- **Niveau de l'OS** : les clients et les fournisseurs sont des processus différents sur la même machine.
  - Le bus CORBA peut alors être tout ou partie du système d'exploitation
- **Niveau du réseau** : les processus sont sur des sites différents et les requêtes sont véhiculées à travers le réseau.
  - C'est le cas sur Internet avec IIOP

L’invocation statique et dynamique

- Ces deux mécanismes complémentaires permettent de soumettre les requêtes aux objets.
- En statique, les invocations sont contrôlées à la compilation.
- En dynamique, les invocations doivent être contrôlées à l’exécution.



L’invocation statique et dynamique

- **Un système auto-descriptif** : les interfaces des objets sont connues du bus et sont aussi accessibles par les programmes par l’intermédiaire du référentiel des interfaces.
- **Activation automatique et transparente des objets** : les objets sont en mémoire uniquement s’ils sont utilisés par des applications clientes.
- **L’interopérabilité entre bus** : à partir de la norme CORBA 2.0, un protocole générique de transport des requêtes (GIOP pour General Inter-ORB Protocol) a été défini permettant l’interconnexion de bus CORBA provenant de fournisseurs distincts
- une de ses instanciations est **l’Internet Inter-ORB Protocol** (IIOP) fonctionnant au dessus de TCP/IP.

OMG Interface Definition Language (OMG IDL)

- Il est utilisé pour **décrire les interfaces des objets distribués**.
- **Séparation de l'interface d'un objet de son implémentation.**
- Il permet de définir dans un langage commun à l'ensemble des compilateurs, les partie visible d'un objet (méthodes , propriétés).
- langage de description d'interfaces totalement indépendant de tout langage de programmation :
- Quelque soit l'implémentation (Java ou C++) le fichier IDL correspondant aux interfaces écrites dans les deux langages sera identique.

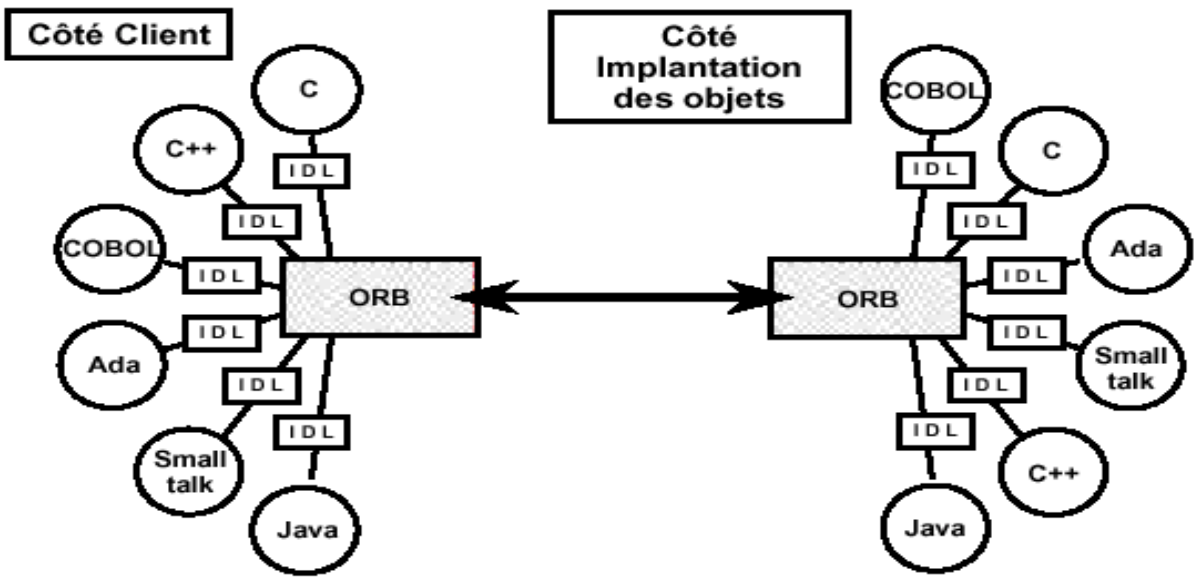
OMG Interface Definition Language (OMG IDL)

- **Les clients interagissent avec les objets locaux et distants en invoquant les méthodes définies dans le fichier IDL**
- IDL est un langage de description (ressemblant au C).
- **Exemple d'interface IDL :**

```
// CompteClient.idl
interface Compte client {
void credit ( in unsigned long montantFRF );
void debit  ( in unsigned long montantFRF );
long solde  ( );
};
```

IDL: PROJECTION

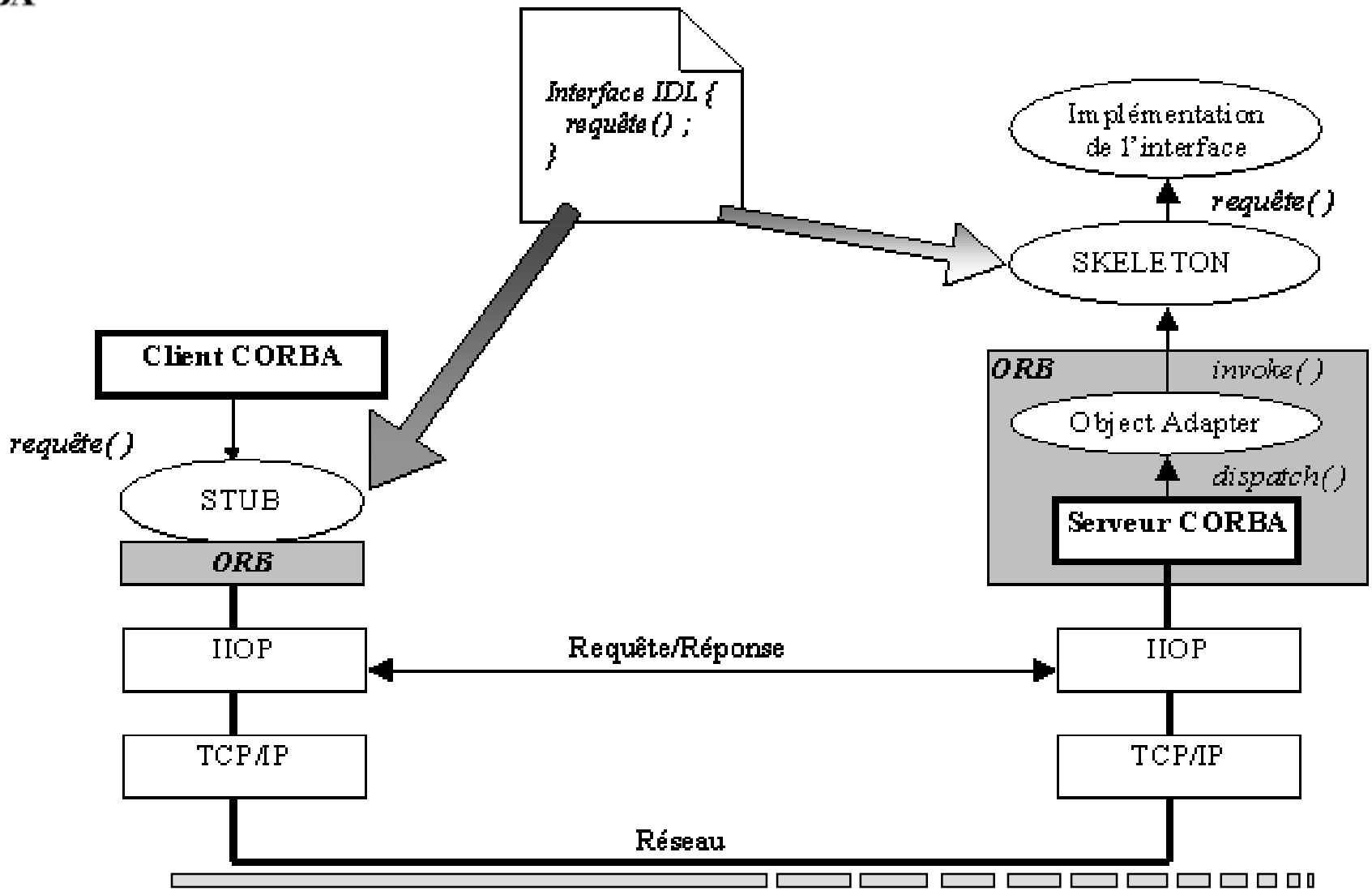
- La projection (ou mapping) : La projection permet de générer du code pour exploiter le type d'objet à partir d'un langage de programmation. Ce mécanisme est dépendant du langage cible.
- Réalisée par un **pré-compilateur IDL** dépendant du langage cible et du bus CORBA cible.



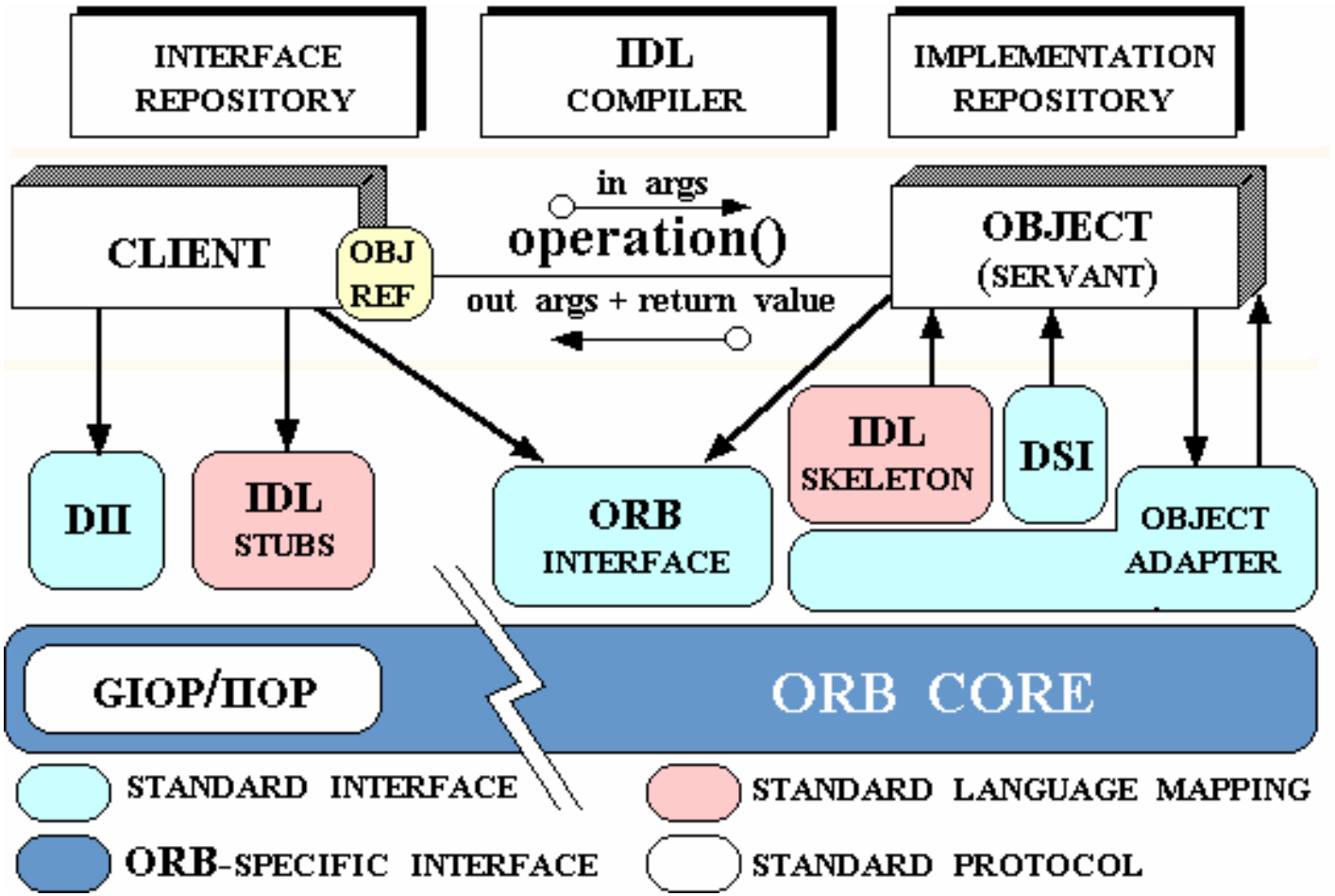
IDL:STUB / SKELETON:

- Avec l'aide d'un compilateur IDL (**IDL COMPILER**) et de vos fichiers idl on génére :
- une souche (**stub**) pour le client un squelette (**skeleton**) pour le serveur.
- On programme l'implémentation de la classe serveur qui doit hériter du squelette généré par idl.
- Une fois les programmes clients et serveurs compilés, l'ORB prendra en charge le pliage/dépliage des requêtes des clients sur les objets locaux et distants.

IDL:STUB / SKELETON:



Les Objets du **Bus d'objets Distribués :** ARCHITECTURE GENERALE



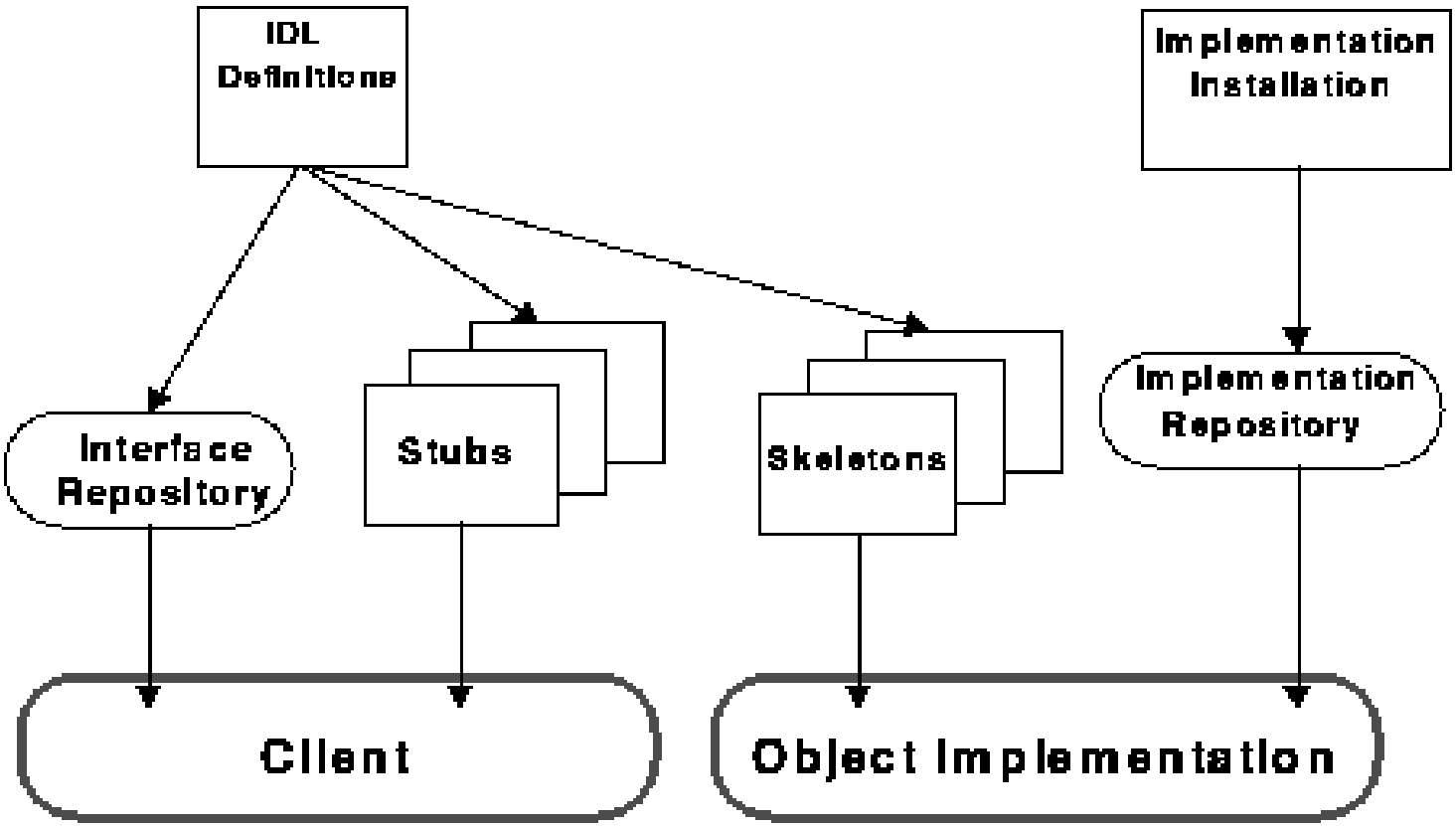
Les Objets du **Bus d'objets Distribués :** Les différents objets

- **Interface Repository ou IR** :c'est le référentiel des interfaces contenant une représentation des interfaces OMG-IDL accessible par les applications durant l'exécution.
- L 'ORB doit avoir accès à la définition de l'objet qu'il gère; cela peut se faire de deux façons:
  - l'information est contenue dans la requête
  - l'ORB prend cette information dans l' interface Repository
- Permet également, en utilisant les informations de l' interface Repository, déterminer à l'exécution les opérations réalisable sur un objet dont l'interface n'était pas connue à la compilation.

Les Objets du **Bus d'objets Distribués :** Les différents objets

- Implementation Repository :Référentiel des implantations
  - Il contient l'information nécessaire à l'activation.
  - Ce référentiel est spécifique à chaque produit CORBA.
- II contient aussi des informations supplémentaires comme par exemple des informations sur le debugging, la sécurité,...

Les Objets du **Bus d'objets Distribués :** Les différents objets



Les Objets du **Bus d'objets Distribués :** Les différents objets

- **OBJ REF :**La référence de l'objet est l'information nécessaire pour désigner sans ambiguïté cet objet dans un ORB.
- Le client et l'implémentation de l'objet ont une notion opaque de la référence de l'objet au travers de leur langage.
- La représentation d'une référence d'objet fournie à un client n'est plus valable à la mort du client.
- Un objet peut avoir plusieurs références d'objet. Deux ORB différents peuvent avoir des représentations de référence d'objet différentes :
- **L'IOR (Interoperable Object Reference)** resoud cette ambiguïté. Il contient notamment l 'adresse machine

Les Objets du **Bus d'objets Distribués :** Les différents objets

- **IDL STUBS :** SII (Static Invocation Interface) : Interface d'invocations statiques
  - permet de soumettre des requêtes contrôlées à la compilation des programmes. Cette interface est générée à partir de définitions OMG-IDL.
- **IDL SKELETON** : SSI (Skeleton Static Interface) Interface de squelettes statiques
  - il permet à l'implantation des objets de recevoir les requêtes leur étant destinées. Cette interface est générée à partir de définitions OMG-IDL

Les Objets du **Bus d'objets Distribués :** Les différents objets

- **DII (Dynamic Invocation Interface)** : Interface d'invocations dynamiques
  - permet de construire dynamiquement des requêtes vers n'importe quel objet CORBA sans générer/utiliser une interface SII.
- **DSI (Dynamic Skeleton Interface) :** Interface de squelettes dynamiques
  - Il qui permet d'intercepter dynamiquement toute requête sans générer une interface SSI.
  - C'est le pendant de DII pour un serveur.

Les Objets du **Bus d'objets Distribués :** Les différents objets

- **Object Adapter** : L'adaptateur d'objets
- " Canal " par lequel l'ORB est connectée à l'implémentation de l'objet.
- Il s'occupe de créer les objets CORBA,
- de maintenir les associations entre objets CORBA et implantations
- de réaliser l'activation automatique si nécessaire.

Les Objets du **Bus d'objets Distribués :** Les différents objets

- les services de l '**'Object Adapter** :
  - ☐génération et interprétation de références d'objet
  - ☐invocation des méthodes
  - ☐sécurité de l'interaction
  - ☐activation et désactivation des objets
  - ☐correspondance des références d'objet aux implémentations
  - ☐liste des implémentations
  - ☐etc...
- la norme CORBA 2.0 introduit le **BOA** ou Basic Object Adapter dépendant du type d 'ORB. Pour résoudre les problèmes d'interopérabilité le **POA** ou *Portable Object Adapter* a été introduit .

Les Objets du **Bus d'objets Distribués :** Les différents objets

- **ORB INTERFACE :** L'interface au bus fournit les primitives de base comme l'initialisation de l'ORB.
  - permet de contrôler le comportement du bus,
  - de créer les autres objets représentant les composantes du bus,
  - de convertir les référence d'objet fournit par le bus en chaîne de caractères et vice-versa
  - et d'obtenir les références des objets « notoires »

Les Objets du **Bus d'objets Distribués :** Les différents objets

- **Object Request Broker (ORB).** Le cœur de CORBA est constitué par l'ORB qui est le conduit/bus par lequel les requêtes sur les objets transitent.
- Noyau de transport des requêtes aux objets.
- Il intègre au minimum les protocoles GIOP et IIOP.
- L'ORB n'est pas directement accessible au programmeur d'applications CORBA qui n'utilise qu'une référence.
-

Les Objets du **Bus d'objets Distribués :** Les différents objets

- Le protocole **GIOP (General Inter-ORB Protocol)** définit :
  - une représentation commune des données CDR ou Common Data Representation
  - un format de références d'objet interopérables (IOR ou Interoperable Object Reference)
  - un ensemble de messages de transport des requêtes aux objets (Request, Reply, …).
  - Cependant, GIOP est seulement un protocole générique

Les Objets du **Bus d'objets Distribués :** Les différents objets

- **IIOP (Internet Inter-ORB Protocol) :**
- IIOP fournit une implantation de GIOP au dessus de TCP/IP et donc d'Internet.
- Les IORs (dans le contexte d'IIOP) doivent contenir :
  - le nom complet de l'interface OMG-IDL de l'objet
  - l'adresse IP de la machine Internet où est localisé l'objet ;
  - un port TCP pour se connecter au serveur de l'objet ;
  - une clé pour désigner l'objet dans le serveur.
- Son format est libre et il est donc différent pour chaque implantation du bus CORBA.

Les différents produits à la norme CORBA

•**VISIBROKER** Commercial CORBA 2.3 Le meilleur produit sur le marché

•**MICO** Gratuit CORBA 2
c'est un produit de très bonne qualité qui peut rivaliser avec des bus commerciaux.

•**ROBIN** gratuit CORBA 2 / freeware de bonne qualité implémenté en Java.

•**ORBACUS** gratuit pour une utilisation non commerciale. CORBA 2

•**OMNIORB2** gratuit. CORBA 2

•**XEROX ILU** gratuit Outil très puissant mais plus complexe à utiliser que les précédents et qui fait partie du projet Inter-Language Unification de Xerox.

•**CORBAScript** gratuit Ce langage interprété permet l'utilisation d'objets CORBA par simples scripts

EN CONCLUSION

- CORBA est une solution ouverte et évolutive avec architecture est modulaire :
- Attention à ne pas tomber dans le pièges tenduspar les fournisseurs de bus : il faut éviter d'utiliser les mécanismes propriétaires d 'un fournisseur pariculier.
- Ces spécifications sont implantées dans des produits « Corba ready ».
- Toujours choisir le produit CORBA le plus approprié à chaque partie d'une application : un produit offrant le POA pour écrire des serveurs C++ portables et un produit offrant la projection IDL/Java pour les applications clientes utilisées à travers le WWW.