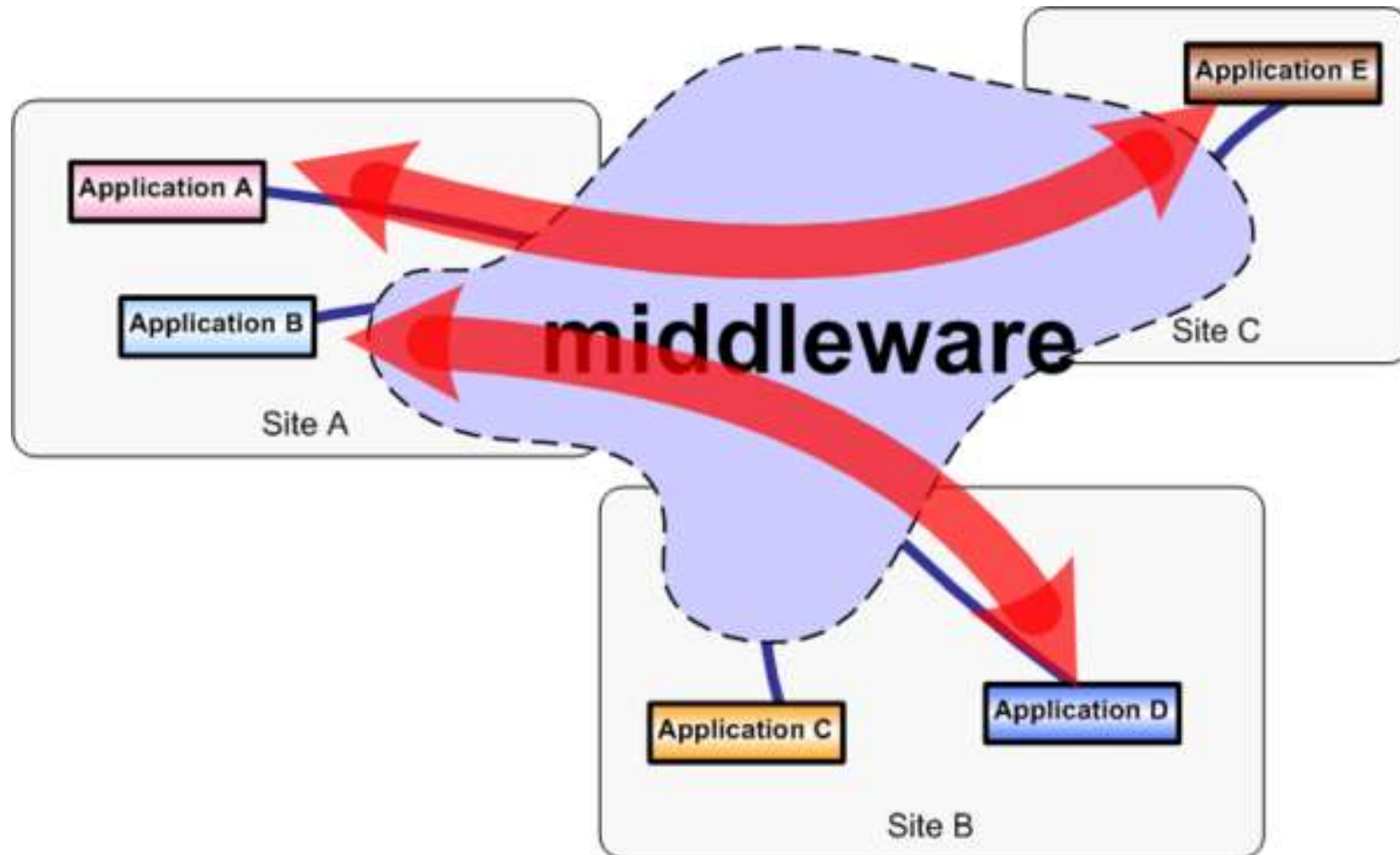


# Middleware, CORBA, une introduction par l'exemple

# ***Points à prendre en compte en amont de la réalisation d'une application distribuée***

1. Protocoles d'accès distants
2. Gestion de la charge
3. Gestion des pannes
4. Persistance
5. Gestion des transactions
6. Programmation multithread
7. Problèmes de nommage
8. Sécurité
9. Cycle de vie des objets
10. Gestion des ressources (Resource pooling)

# *Middleware*



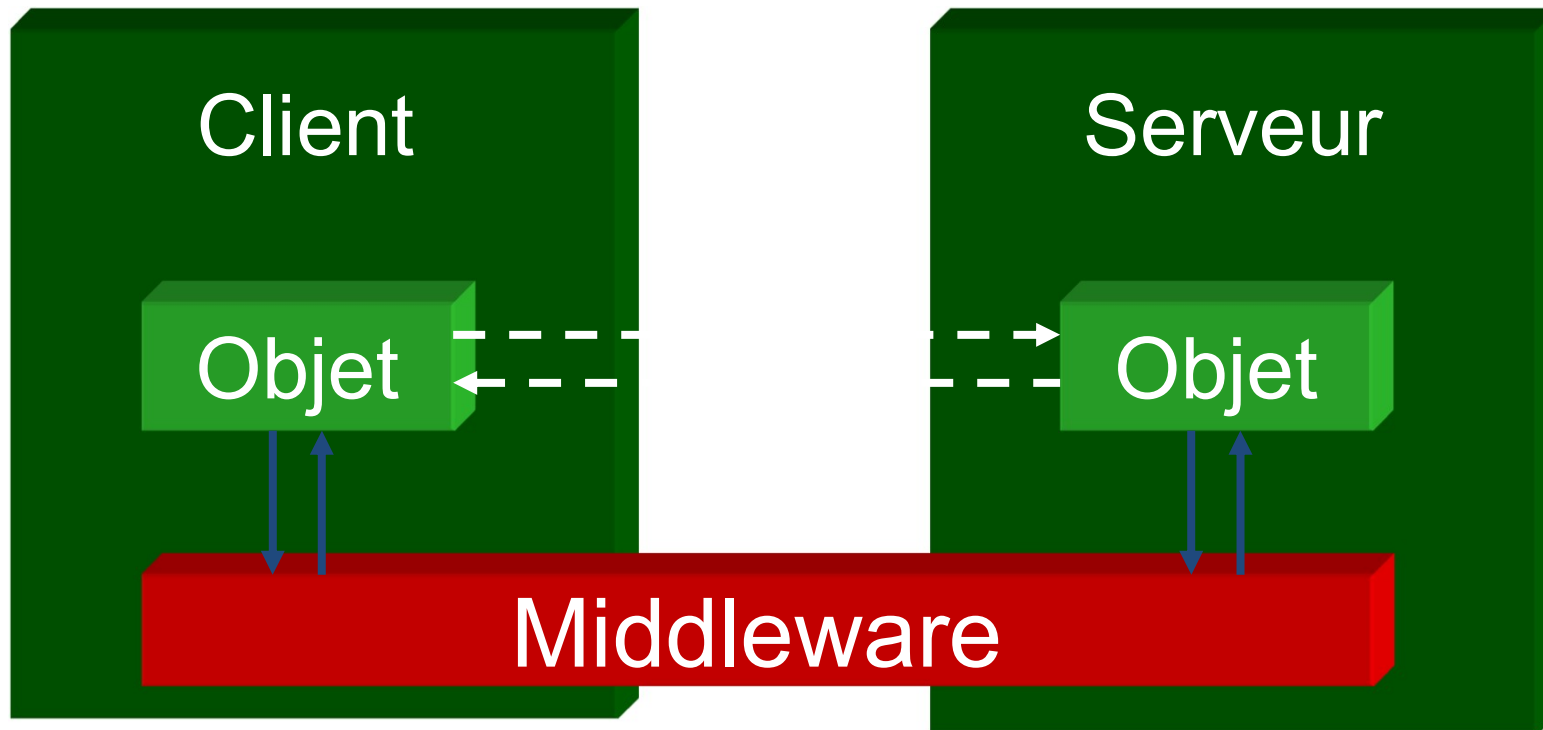
# Qu'est ce que C.O.R.B.A. ?

- ▶ **CORBA** ( Common Object Request Broker Architecture ) est une architecture logicielle, pour le développement de composants.
- ▶ Ces composants peuvent être écrits dans des langages de programmation (objets) distincts, être exécutés dans des processus séparés, voire être déployés sur des machines distinctes. On parle ainsi **d'architectures réparties ou distribuées**.
- ▶ L'architecture CORBA a été créée par l'OMG en 1992.

# Qu'est ce que C.O.R.B.A. ?

- ▶ **CORBA** ( Common Object Request Broker Architecture ) est une architecture logicielle, pour le développement de composants.
- ▶ Ces composants peuvent être écrits dans des langages de programmation distincts, être exécutés dans des processus séparés, voire être déployés sur des machines distinctes.
- ▶ On parle ainsi **d'architectures réparties ou distribuées**.
- ▶ La norme CORBA a été créée par l'OMG en 1992.

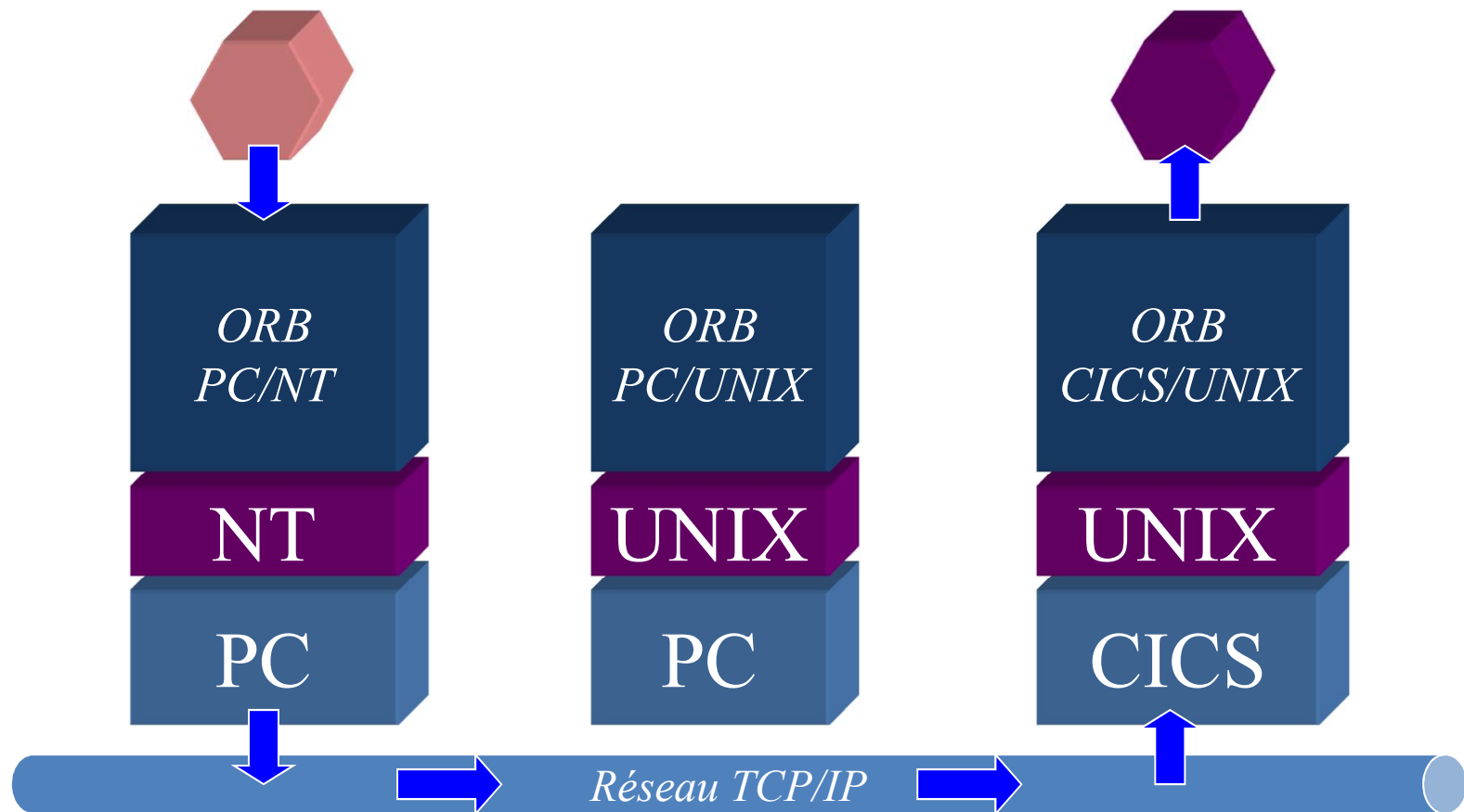
# ***CORBA - Principe simplifié***



# Objectifs de CORBA

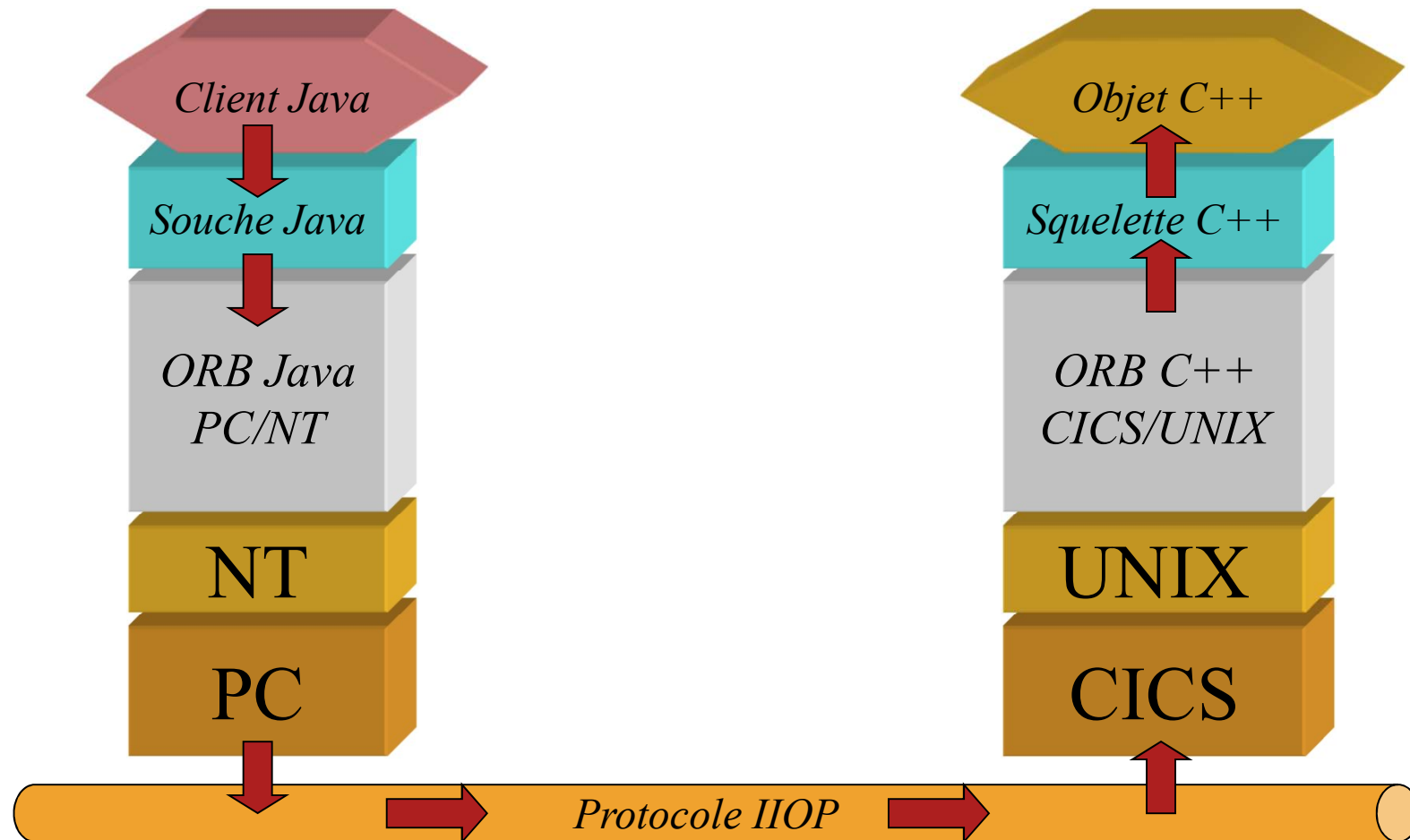
- ▶ Fournir un environnement ouvert
  - Les membres participent aux spécifications de la norme
  
- ▶ Fournir un environnement portable
  - Les API sont définies pour rendre les applications portables, et ce, quelque soit le produit CORBA utilisé.
  
- ▶ Fournir un environnement interopérable
  - Permettre aux applications CORBA de collaborer entre elles.

## *Le dénominateur commun : Le bus CORBA*

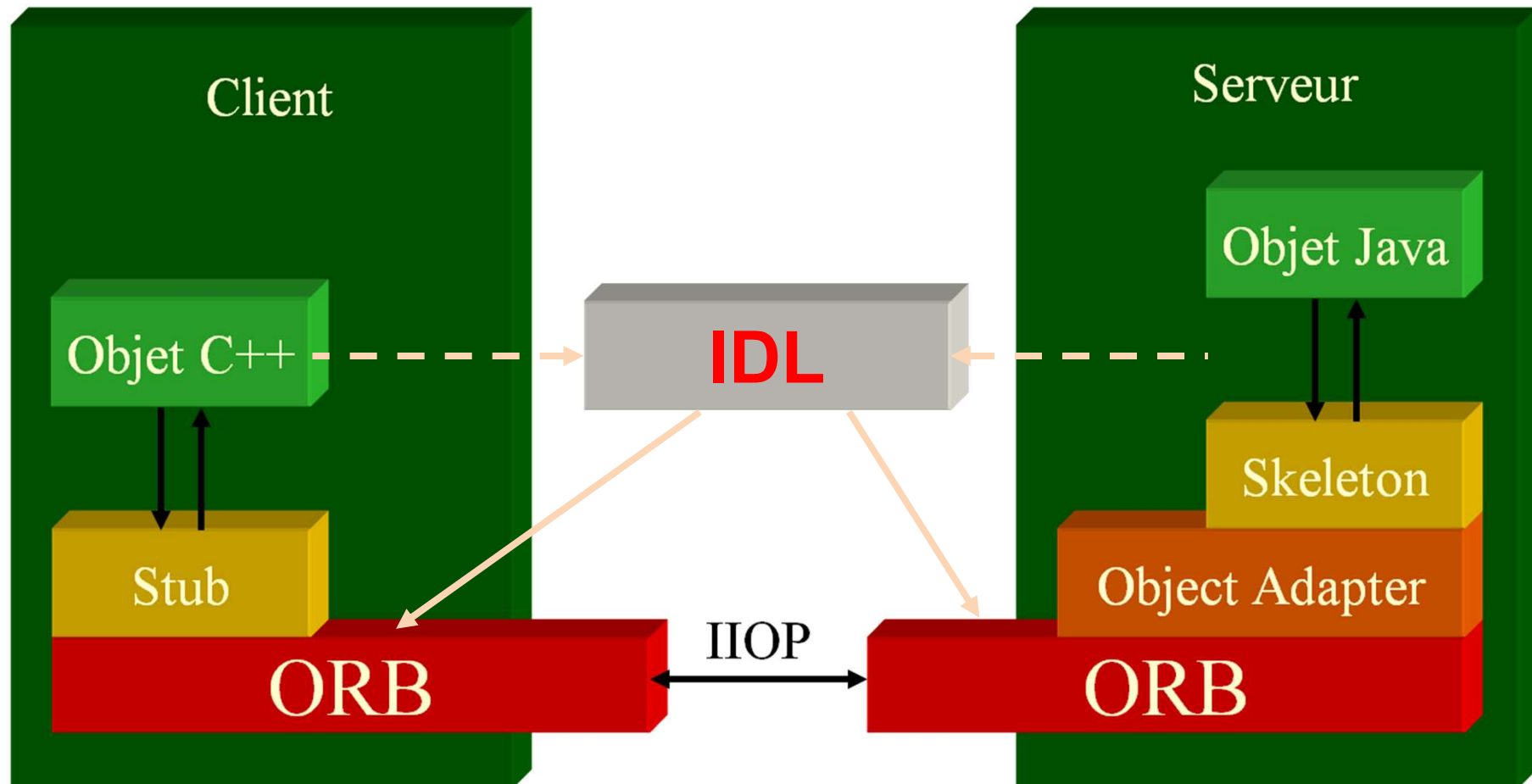




## *Intervention des amorces (souche/squelette) CORBA*





# Architecture de CORBA



# ***Serveur et objets***

- ▶ Un serveur CORBA peut héberger plusieurs objets CORBA.
- ▶ Chaque objet est accessible indépendamment des autres objets du serveur.
- ▶ Chaque objet exprime son offre de services. Pour cela, on utilise un langage de description de services appelé IDL.

# Composants

- ▶ **Stub** : partie cliente
  - ▶ **Skeleton** : partie serveur
- 
- générés  
automatiquement
- ▶ **Object Adapter** : enregistrement des objets (attribution d'un identifiant à chaque objet)
- 
- ▶ **ORB** (Object Request Bus)
  - ▶ **IIOP** (Internet Inter-ORB Protocol)
- 
- transport des  
messages

# ***Le langage IDL CORBA***

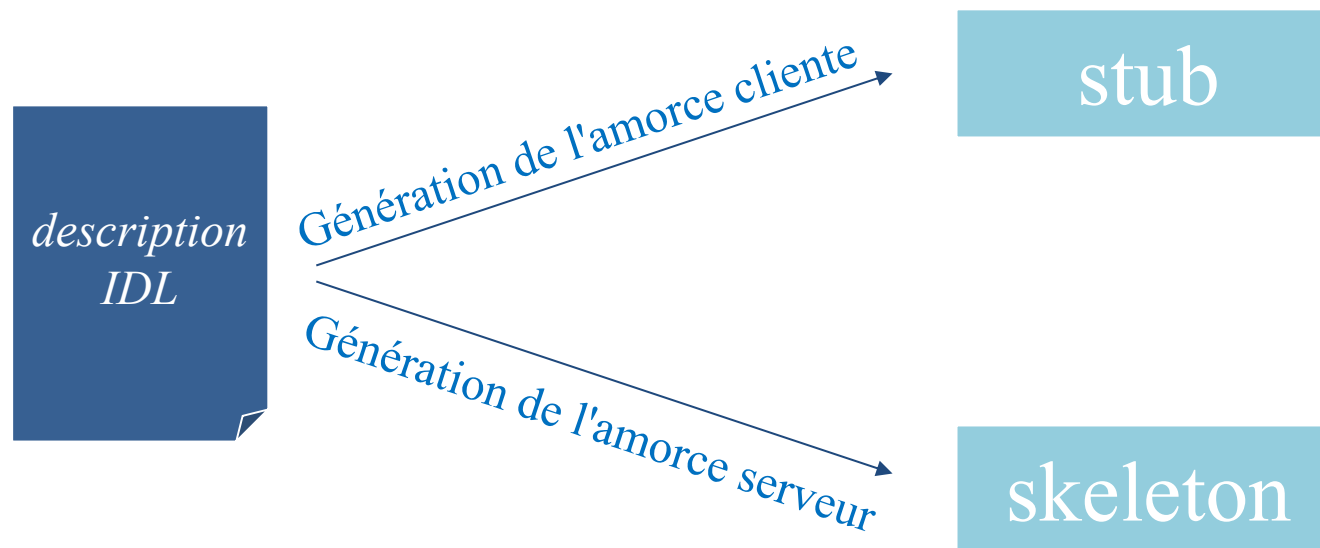
- Il s'agit de décrire au sein d'une interface la liste des services offerts, dans un langage à part entière.

## *Exemple*

```
interface Horloge{  
    string donne_heure_a_paris();  
    string donne_heure_a_pekin();  
};
```

# ***La compilation IDL***

- Une description IDL est compilée pour générer les amorces nécessaires au mécanisme RPC : stub et skeleton



# ***Stub : Coté client***

## ▶ **Fonctions du stub**

- Prépare les paramètres d'entrée de l'invocation
- Décode les paramètres de sortie et le résultat

## ▶ **Stub statique**

- Une par type d'objet serveur à invoquer
- Identique aux talons clients RPC
- Générée à la compilation à partir de l'interface IDL

## ▶ **Stub dynamique**

- Souche générique construisant dynamiquement tout type de requêtes
- Permet d'invoquer des objets serveurs que l'on découvre à l'exécution (i.e. dont on ne connaît pas l'interface à la compilation : Référentiel d'interfaces)

# ***Skeleton : Côté serveur***

## ▶ **Fonctions du skeleton**

- décode les paramètres d'entrée des invocations
- prépare les paramètres de sortie et le résultat

## ▶ **Skeleton statique**

- un par type d'objet serveur invocable
- identique aux talons serveurs RPC
- généré à la compilation à partir de l'interface IDL

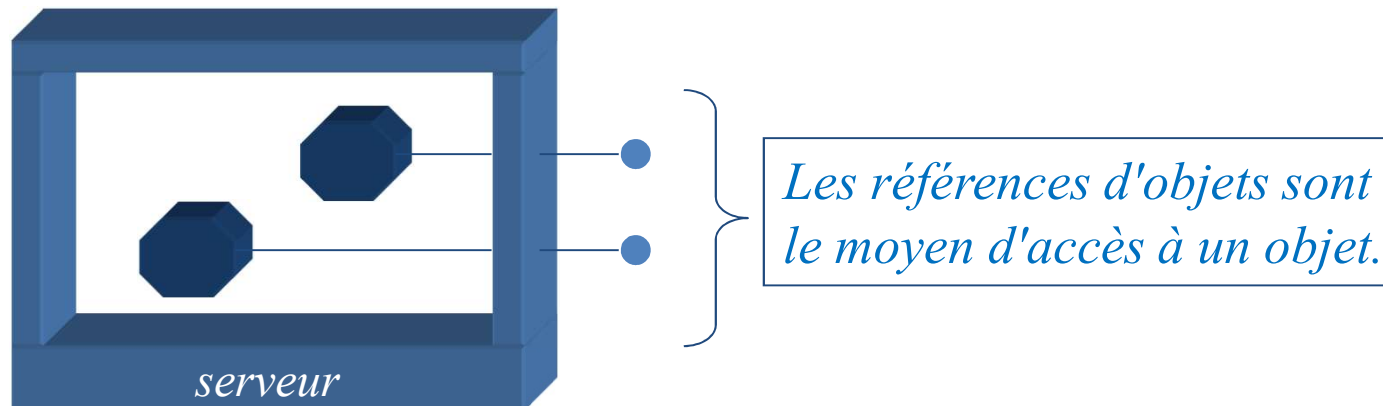
## ▶ **Skeleton dynamique**

- squelette générique prenant en compte dynamiquement tout type de requêtes
- permet de créer à l'exécution des classes d'objets serveurs (i.e. que l'on ne connaissait pas à la compilation)

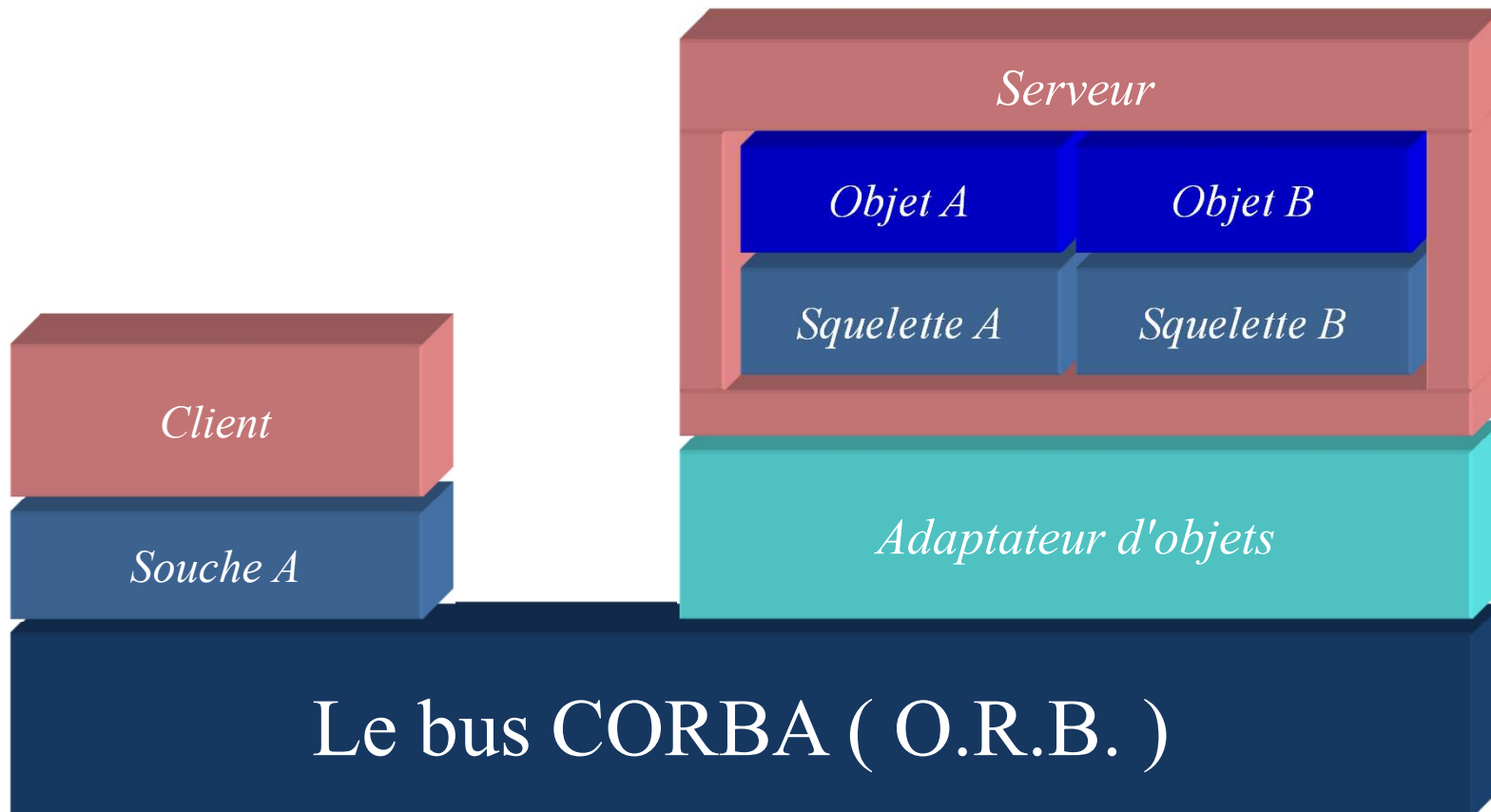


# ***L'identité d'un objet C.O.R.B.A.***

- ▶ Chaque objet C.O.R.B.A. est associé à une référence d'objet qui forme son "identité".
- ▶ Deux objets C.O.R.B.A. du même type ont deux identités différentes.



## *L'adaptateur d'objets*



# ***L'adaptateur d'objets***

## **► Fonctions**

- Interface entre les objets CORBA et l'ORB
- Enregistrement et recherche des implantations d'objets
- Génération de références pour les objets
- Gestion de l'instanciation des objets serveurs
- Activation des processus dans le serveur
- Aiguillage des invocations de méthodes vers les objets serveurs

## **► Différents type d'adaptateur**

- BOA (Basic Object Adapter)
- POA (Portable Object Adapter)

# ***Les communications avec CORBA***

- ▶ Les participants à un échange CORBA communiquent à l'aide d'un protocole spécifique à CORBA : **IOP** ( Internet Inter-ORB Protocol ).
- ▶ Le protocole IOP est indépendant du langage de programmation, du système d'exploitation et de la machine utilisée.

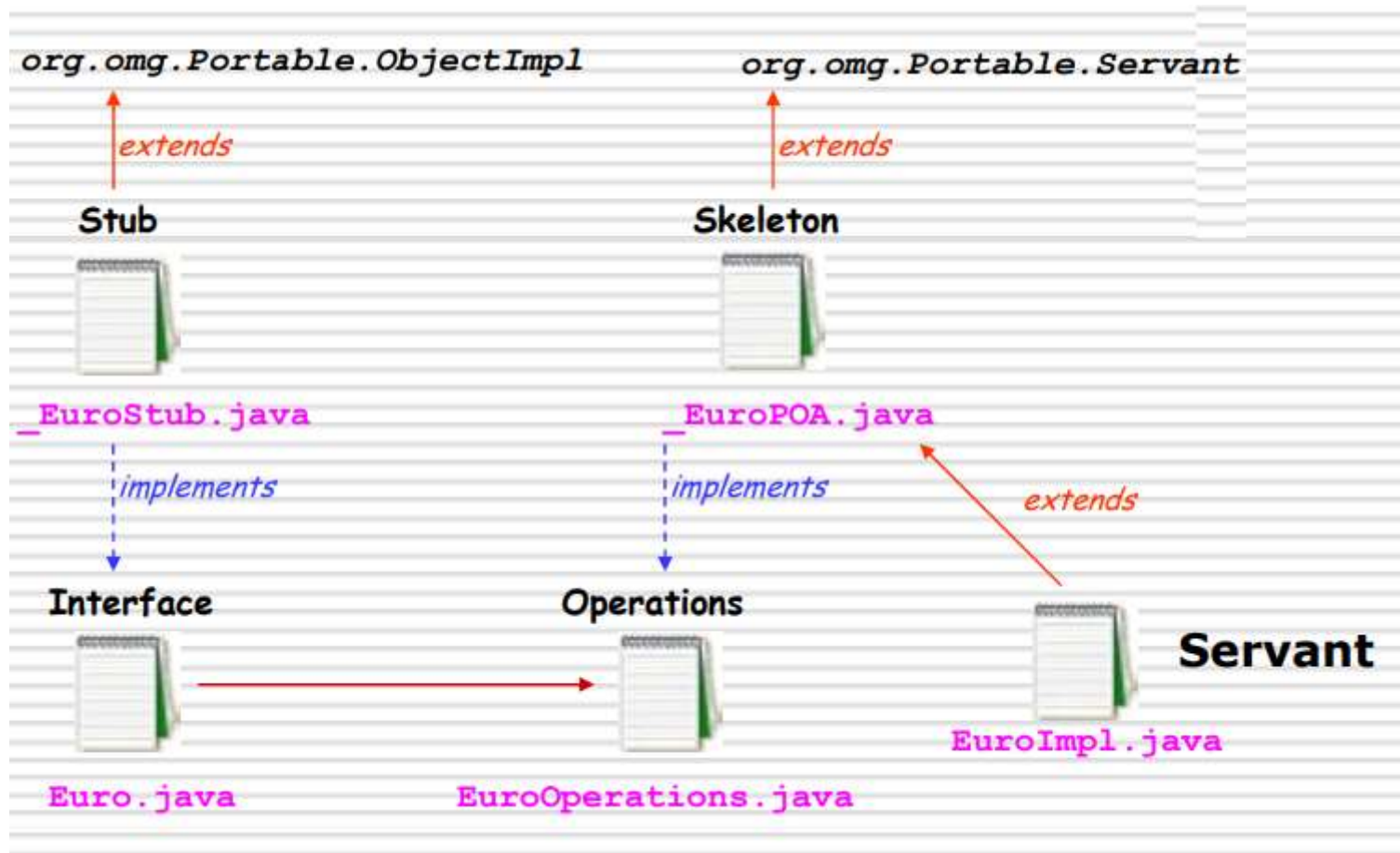
# ***Les services C.O.R.B.A.***

- ▶ Pour accélérer et faciliter le développement d'applications avec C.O.R.B.A., l'O.M.G a spécifiée un ensemble de services.
- ▶ A l'heure actuelle, ***plus de 17 services*** ont été définis.
- ▶ Les services ***sont vendus séparément*** du bus CORBA.
- ▶ Seul quelques services sont actuellement disponibles sur le marché

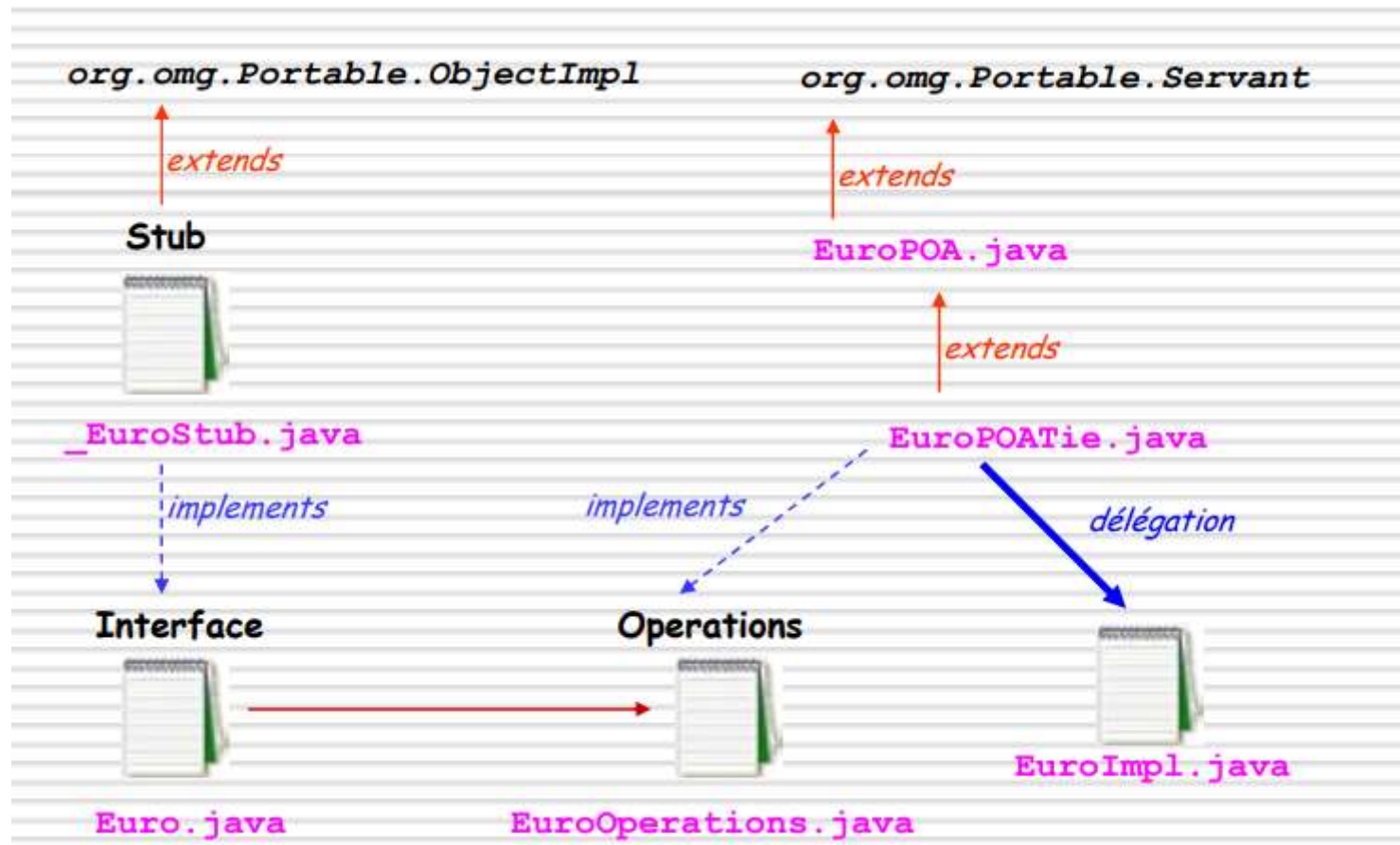
# ***L'annuaire C.O.R.B.A.***

- ▶ L'annuaire C.O.R.B.A. est un service.
- ▶ Il s'agit d'un serveur qui enregistre des associations nom / référence d'objet.
- ▶ Un serveur peut enregistrer ses objets dans l'annuaire.
- ▶ Un client peut récupérer l'accès à un objet en consultant l'annuaire.

# Mise en œuvre par héritage



# Mise en œuvre par délégation





# ***La description IDL***

- ▶ La description doit être compilée afin de générer les amorces ( souche et squelette ) requises pour l'établissement de la communication inter-processus.
- ▶ A l'issu de la compilation, plusieurs fichier sont créés :
  - Le squelette
  - La souche,
  - L'interface
  - Les opérations de l'interface

## ***Concept de « mapping »***

- ▶ Une description IDL est traduite vers un langage de programmation.
- ▶ Les règles de traduction sont appelées "mapping" et font partie de la spécification CORBA.
- ▶ Grâce au mapping, deux fournisseurs d'ORBs offriront le même modèle de programmation.

# Mise en oeuvre

- ▶ La mise en œuvre d'une application CORBA respecte les étapes suivantes :

1. Description des objets à l'aide de l'IDL
2. Compilation de la description IDL
3. *Génération des divers objets (helpers)*
4. Création de l'objet serveur (+ compilation)
5. Création de l'objet client. (+ compilation)
6. Démarrage de l'ORB

## ***Conclusions***

- ▶ Solution non propriétaire => Standard
- ▶ Libre choix de l'implémentation et du langage
- ▶ Solution fonctionnelle d'interopérabilité
- ▶ Plusieurs implémentations libres et open-source (MICO Corba, OpenFusion,...)
- ▶ **Mais.. Mise en œuvre assez complexe..**

# ***DCOM***

- ▶ **DCOM** (Distributed Component Object Model) une architecture propriétaire de Microsoft qui permet la communication entre des composants logiciels distribués.
- ▶ DCOM est l'alternative de Microsoft à l'architecture CORBA.
- ▶ *DCOM a été rendu obsolète par l'avènement de la plateforme .Net*