Chapitre III

Intergiciels orientés objets (CORBA)

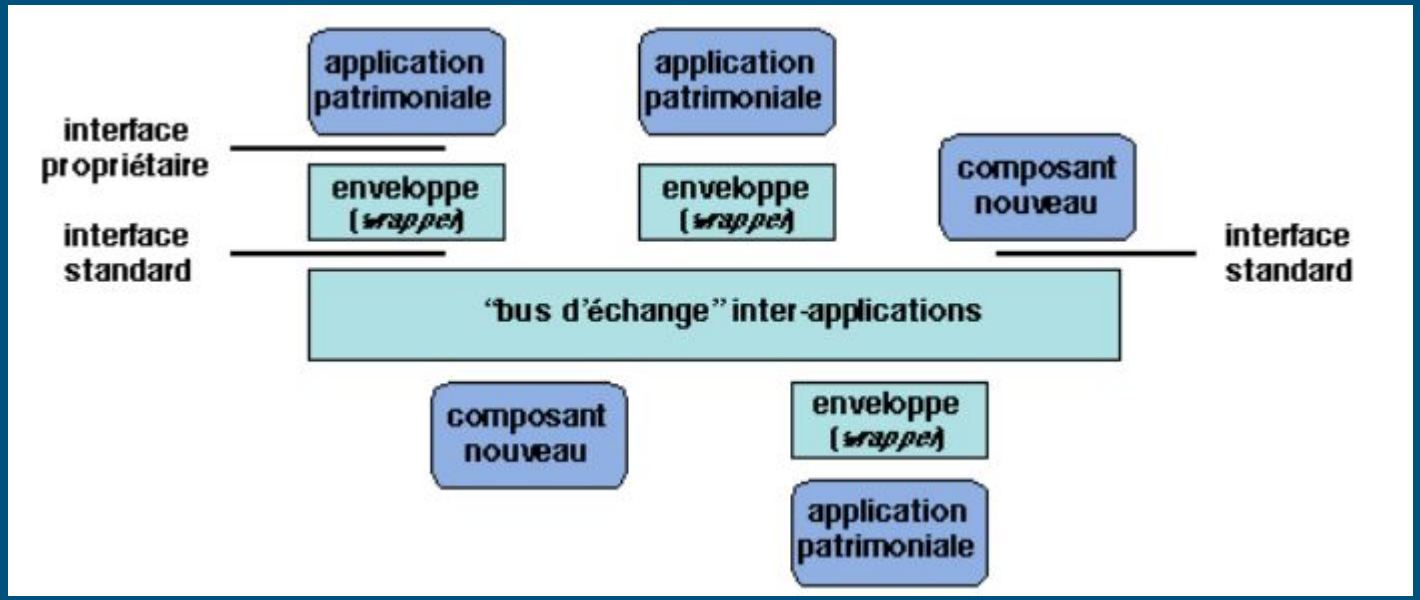
I. Introduction aux intergiciels

- **La modernisation des applications patrimoniales** (transformation /conversion d'un système patrimonial en une infrastructure moderne) est un objectif identifié du génie logiciel.
- Une application patrimoniale ne peut être utilisée qu'à travers une interface spécifiée, et ne peut être modifiée. La plupart du temps, l'application doit être reprise telle quelle car le coût de sa réécriture serait trop élevé.
- Le principe des solutions actuelles est d'adopter une norme commune, non liée à un langage particulier, pour interconnecter différentes applications.

I. Introduction aux intergiciels

- La norme spécifie des interfaces et des protocoles d'échange pour la communication entre applications. Les protocoles sont réalisés par une couche logicielle qui fonctionne comme **un bus d'échanges** entre applications (en anglais broker).
- Pour intégrer une application patrimoniale, il faut développer une enveloppe (en anglais wrapper), c'est-à-dire une couche logicielle qui fait le pont entre l'interface originelle de l'application et une nouvelle interface conforme à la norme choisie.

I. Introduction aux intergiciels

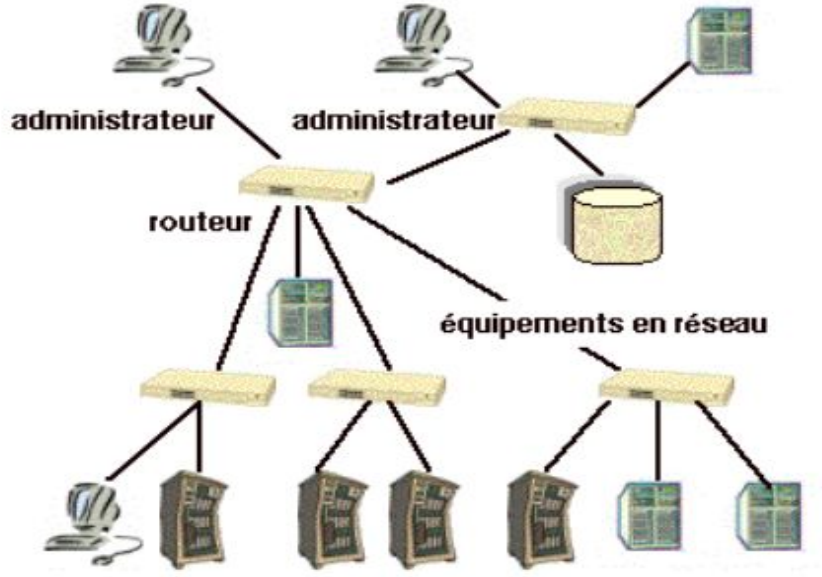


Intégration d'applications patrimoniales

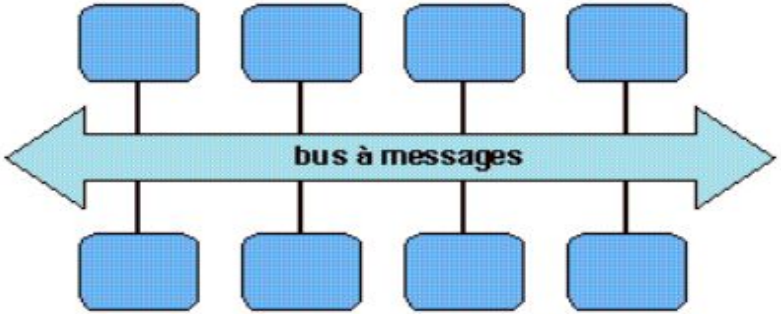
I. Introduction aux intergiciels

- Une application patrimoniale ainsi «enveloppée» peut maintenant être intégrée avec d'autres applications du même type et avec des composants nouveaux, en utilisant les protocoles normalisés du courtier. Des exemples de courtiers sont CORBA, les files de messages, les systèmes à publication et abonnement (en anglais publish-subscribe)
- **Un courtier de messagerie** consiste en **un programme intermédiaire** qui traduit les messages du protocole de messagerie formel de l'expéditeur vers le protocole de messagerie formel du destinataire. Les programmes de courtage de messagerie sont parfois connus sous le nom de logiciels intermédiaires, ou « middleware »

I. Introduction aux intergiciels



(a) Organisation physique



(b) Organisation logique

Surveillance et commande d'équipements en réseau

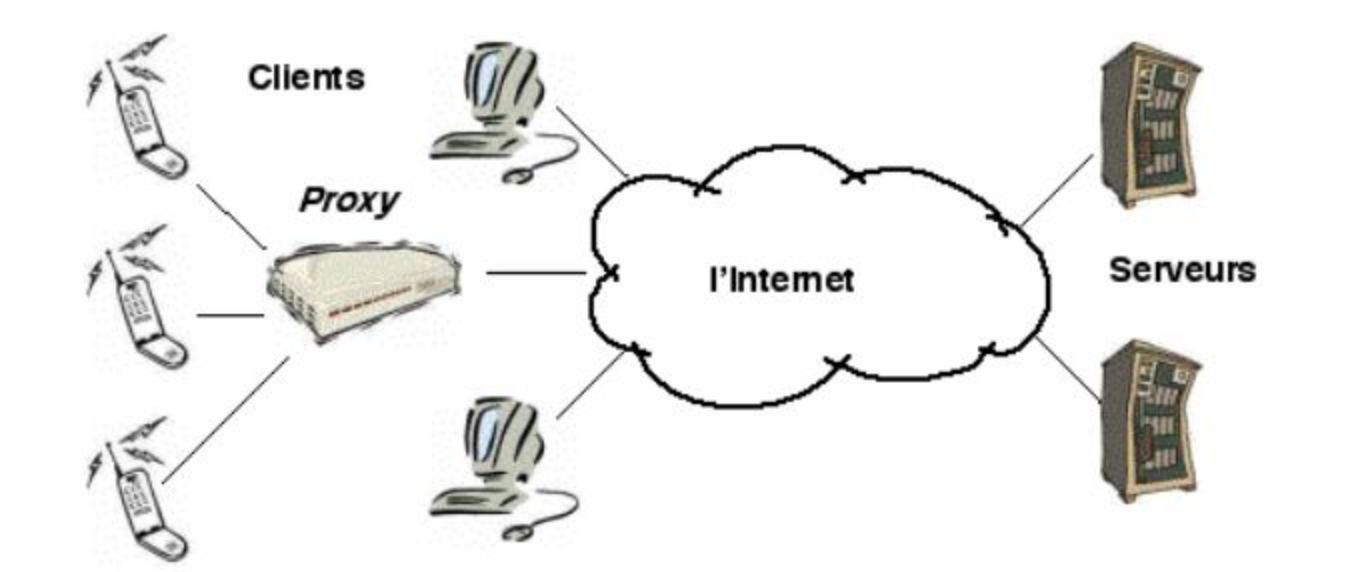
I. Introduction aux intergiciels

- La gestion de tels systèmes comporte des tâches telles que la mesure continue des performances, l'élaboration de statistiques d'utilisation, l'enregistrement d'un journal, le traitement des signaux d'alarme, la collecte d'informations pour la facturation, la maintenance à distance, le téléchargement et l'installation de nouveaux services.
- L'exécution de ces tâches nécessite l'accès à distance au matériel, la collecte et l'agrégation de données, et la réaction à des événements critiques.
- Les systèmes réalisant ces tâches s'appellent **systèmes de médiation**.

I. Introduction aux intergiciels

- L'infrastructure interne de communication d'un **système de médiation** doit réaliser la collecte de données et leur acheminement depuis ou vers les capteurs et effecteurs. La communication est souvent déclenchée par un événement externe, comme la détection d'un signal d'alarme ou le franchissement d'un seuil critique par une grandeur observée.
- Un système de communication adapté à ces exigences **est un bus à messages**, c'est-à-dire un canal commun auquel sont raccordées les diverses entités.

I. Introduction aux intergiciels



Adaptation des communications aux ressources des clients
par des **mandataires**

I. Introduction aux intergiciels

- Entre un PC de haut de gamme, un téléphone portable et un assistant personnel, les écarts de bande passante, de capacités locales de traitement, de capacités de visualisation, sont très importants.
- On ne peut pas attendre d'un serveur qu'il adapte les services qu'il fournit aux capacités locales de chaque point d'accès. On ne peut pas non plus imposer un format uniforme aux clients.
- La solution préférée est d'interposer **une couche d'adaptation**, appelée **mandataire** (en anglais proxy) entre les clients et les serveurs.

I. Introduction aux intergiciels

- **Un mandataire différen**t peut être réalisé pour chaque classe de point d'accès côté client (téléphone, assistant, etc.).
- **La fonction du mandataire est d'adapter les caractéristiques** du flot de communication depuis ou vers le client aux capacités de son point d'accès et aux conditions courantes du réseau.

➔ Les trois exemples qui précèdent ont une caractéristique commune : dans chacun des systèmes présentés, des applications utilisent **des logiciels de niveau intermédiaire**, installés au-dessus des systèmes d'exploitation et des protocoles de communication, qui **réalisent les fonctions suivantes** :

I. Introduction aux intergiciels

1. **Cacher la répartition**, c'est-à-dire le fait qu'une application est constituée de parties interconnectées s'exécutant à des emplacements géographiquement répartis ;

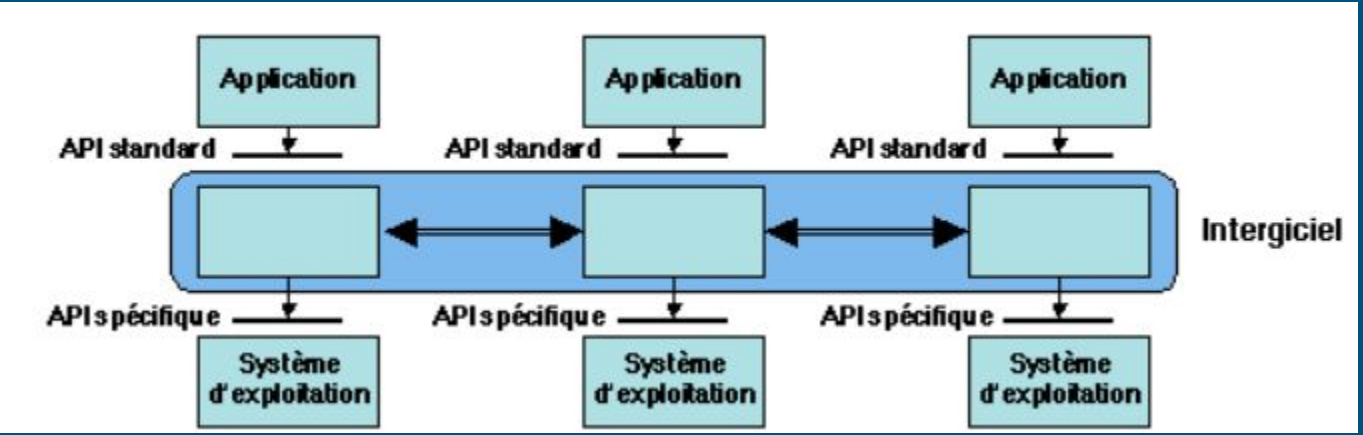
2. **Cacher l'hétérogénéité** des composants matériels, des systèmes d'exploitation et des protocoles de communication utilisés par les différentes parties d'une application ;

3. **Fournir des interfaces uniformes, normalisées,** et de haut niveau aux équipes de développement et d'intégration, pour faciliter la construction, la réutilisation, le portage et l'interopérabilité des applications ;

I. Introduction aux intergiciels

- Un middleware, appelé aussi logiciel médiateur ou intergiciel, se présente sous la forme d'un logiciel. Il constitue une couche technique supplémentaire entre le système d'exploitation (OS, Operating System) et les applications afin de faciliter leurs interactions.
- Le middleware permet également la communication de données entre applications hétérogènes.
- Un intergiciel peut être à usage général ou dédié à une classe particulière d'applications

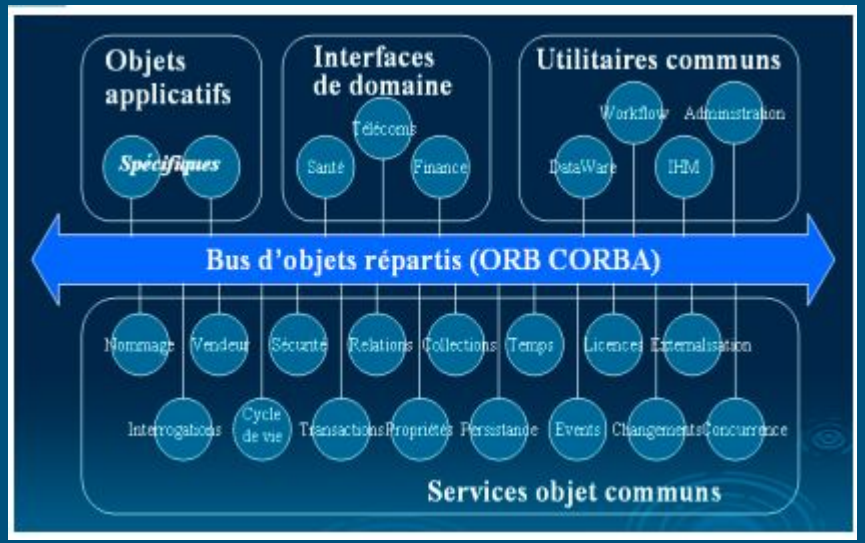
I. Introduction aux intergiciels



Organisation de l'intergiciel

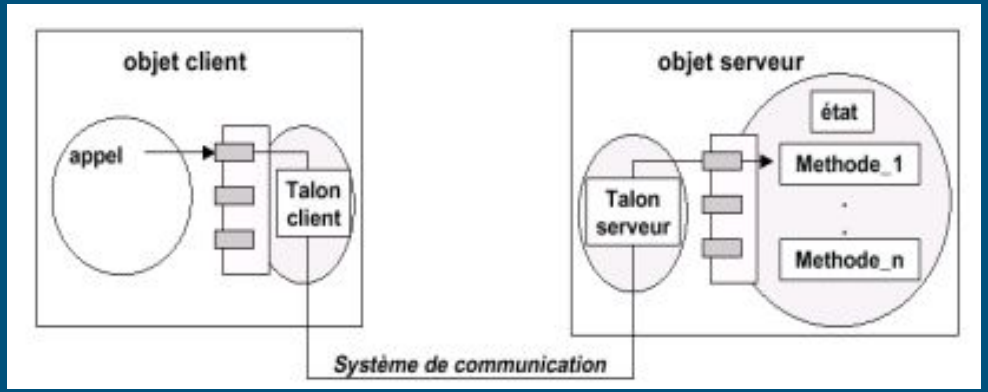
II. Architecture OMA / CORBA

CORBA (Common Object Request Broker Architecture) est un standard décrivant l'architecture suivante :



II. Architecture OMA / CORBA

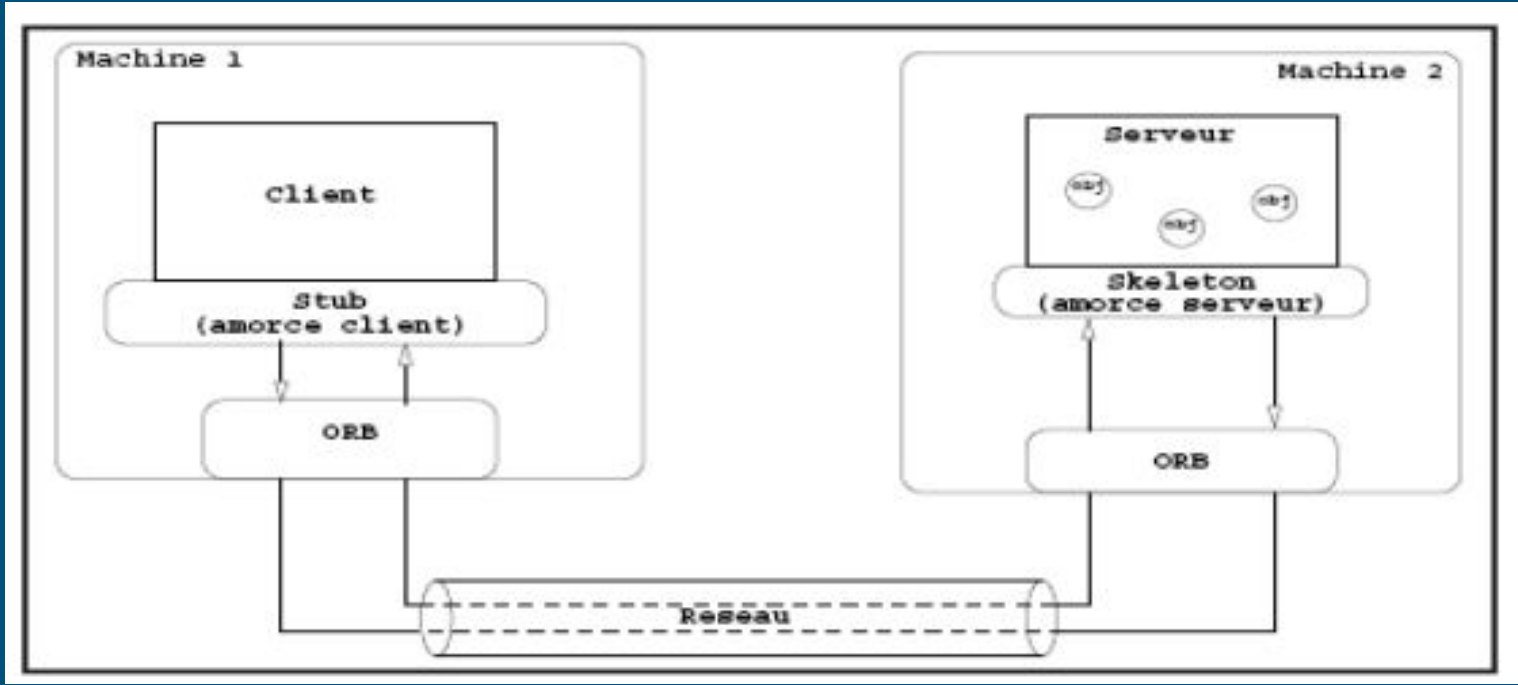
Elle a été spécifiée pour la première fois par l'OMG (Object Management Group) en 1992. CORBA est basée sur un bus, l'ORB (Object Request Broker), qui assure les collaborations entre applications).



II. Architecture OMA / CORBA

- **Comme pour les RMIs les communications sont basées sur le mécanisme d'invocation de procédure distantes et requièrent la création d'amorces qui se branchent au bus et permettent l'émission et la réception de messages entre les clients et les serveurs.**
- **L'ORB prend généralement la forme d'une bibliothèque de fonctions assurant la communication entre les clients et les serveurs.**
- **CORBA est une spécification et non un langage**

II. Architecture OMA / CORBA



Mécanisme d'invocation de procédures distantes

III. Composants et services CORBA Langage IDL et projection en JAVA

Le rôle d'un serveur est de mettre un ensemble d'objets à la disposition des clients. Pour pouvoir accéder à ces objets, les clients doivent pouvoir connaître l'ensemble des méthodes qu'ils peuvent invoquer sur ces objets. Ceci est fait par l'intermédiaire de "contrats" définis à l'aide de l'Interface Definition Language (IDL).
Le langage IDL permet d'exprimer la coopération entre les fournisseurs et les utilisateurs de services en séparant l'interface de l'implémentation.

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

III. Composants et services CORBA Langage IDL et projection en JAVA

- **Le contrat entre les fournisseurs et les clients s'exprime sous la forme d'un ensemble d'interfaces spécifiées à l'aide du langage IDL. Une interface décrit l'ensemble des opérations fournies par un type d'objet CORBA.**
- **La notion d'interface est similaire à celle de classe utilisée en programmation orientée objet. Une interface met en œuvre des méthodes et des attributs dont il est nécessaire de définir le type.**
- **CORBA étant destiné à créer des applications interopérables, les types de base utilisés sont spécifiques au langage IDL (il ne s'agit ni de types Java, ni de types C++,...). Ces types sont ensuite projetés sur les langages dans lesquels sont réalisées les implémentations.**

III. Composants et services CORBA Langage IDL et projection en JAVA

IDL	Java
module	package
interface	Interface
operation	method
exception	exception

Type IDL	Type Java
boolean	boolean
char/ wchar	char
octet	byte
short / unsigned short	short
long / unsigned long	int
long long / unsigned long long	long
float	float
double	double
string / wstring	string

Correspondances entre le langage IDL et java

III. Composants et services CORBA Langage IDL et projection en JAVA

Le but du serveur est de mettre des objets (correspondant à l'implémentation) à la disposition de ses clients, de recevoir les requêtes. Son fonctionnement se déroule en 6 étapes :

- **initialisation de l'ORB**
- **Création de l'objet**
- **Activation de l'objet**
- **Enregistrement de l'objet auprès du BOA**
- **Mise en attente des requêtes des clients**
- **Le lancement du serveur**

III. Composants et services CORBA Langage IDL et projection en JAVA

Le but du client est d'accéder à l'objet distant et d'invoquer les méthodes proposées par cet objet. Son fonctionnement se déroule en 5 étapes :

- **initialisation de l'ORB**
- **Obtention d'une référence à l'objet distant**
- **Extraction de l'objet (lien au stub) correspondant**
- **Invocation de la méthode distante**
- **Le lancement du client**

III. Composants et services CORBA Langage IDL et projection en JAVA

- **CORBA propose la possibilité d'accéder à un objet distant à partir d'un nom plus facile à mémoriser que la référence retournée par la méthode object_to_string.**
- **La mise en correspondance d'un nom et d'un objet se fait grâce au service de nommage.**
- **Avec Java, le service de nommage est lancé à l'aide de la commande : tnameserv -ORBInitialPort port**

Fin

Merci Pour Votre Attention