INTRODUCTION GENERALE
UE NSY014

Applications Réparties

Environnement de l'enseignement

- Yann POLLET : Professeur Titulaire de la Chaire INTEGRATION DES SYSTEMES

- Yves LALOUM : Professeur Associé au CNAM Chaire Intégration des Systèmes

OBJECTIFS DE L'UE

- Objectif 1:
  - Donner une vision claire de la répartition de services et de données à travers l'ensemble des technologies existantes et coopérantes en entreprise
- Objectif 2 :
  - Introduire les outils permettant de concevoir des systèmes et des applications réparties
- Objectif 3 :
  - Préparer à d'autres valeurs du CNAM plus spécialisées sur les technologies.

Forme de l'enseignement

- semestre de cours TP et TD
  - Les mardi soirs de 18H00 à 21H30
  - Séances de cours, TD et TP.

Validation de l'enseignement

- 2 sessions d'examen :
  - Au mois de juin
  - Au mois de Septembre en rattrapage

Préparation à l'examen :

  - Suivi des cours et supports de cours publiés sur le site du CNAM au fur et à mesure (http://deptinfo.cnam.fr)
  - Bibliographie recommandée au niveau de chaque cours
  - Autres…

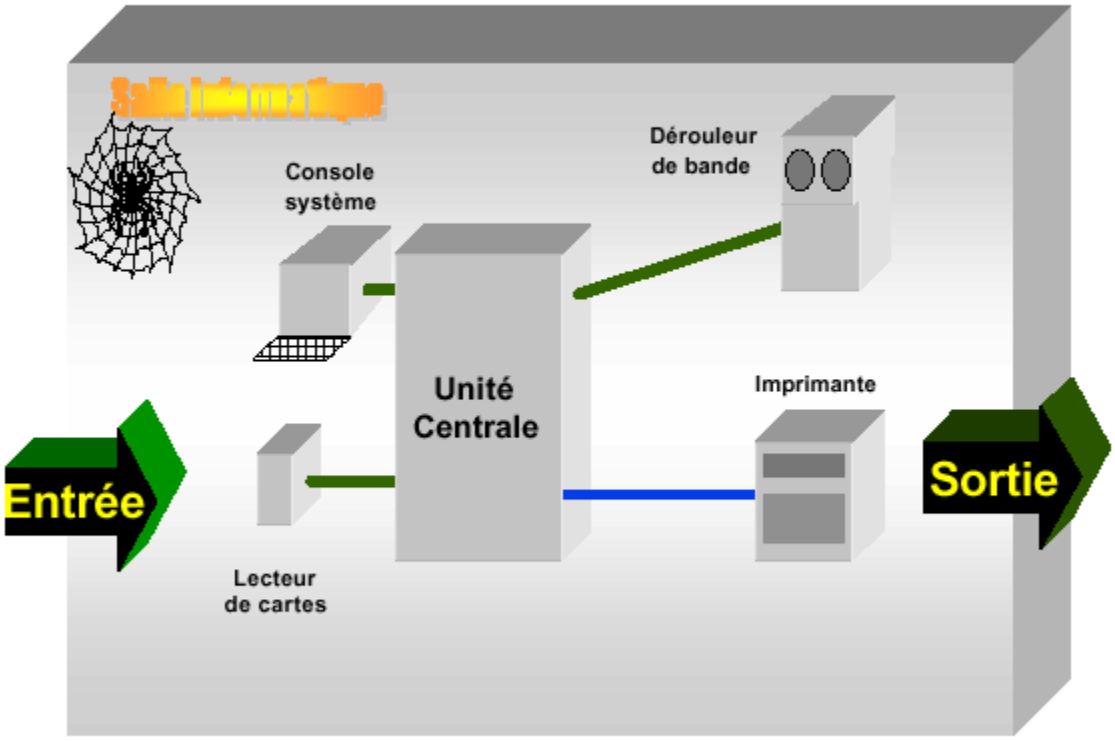
- Exposés à faire dans le cadre de l'enseignement
- Note requise pour obtention de l'UE: 10

Equipe Pédagogique

- Yves LALOUM : Responsable de l'UE
  Professeur Associé
- Jean Marc FARINONE : Maître de Conférence
- Guillaume LEVIEUX : Maître de Conférence
- Djamel BELLEBIA : Responsable de développement extérieur

Au commencement….

1. Début de l'informatique : années 1960

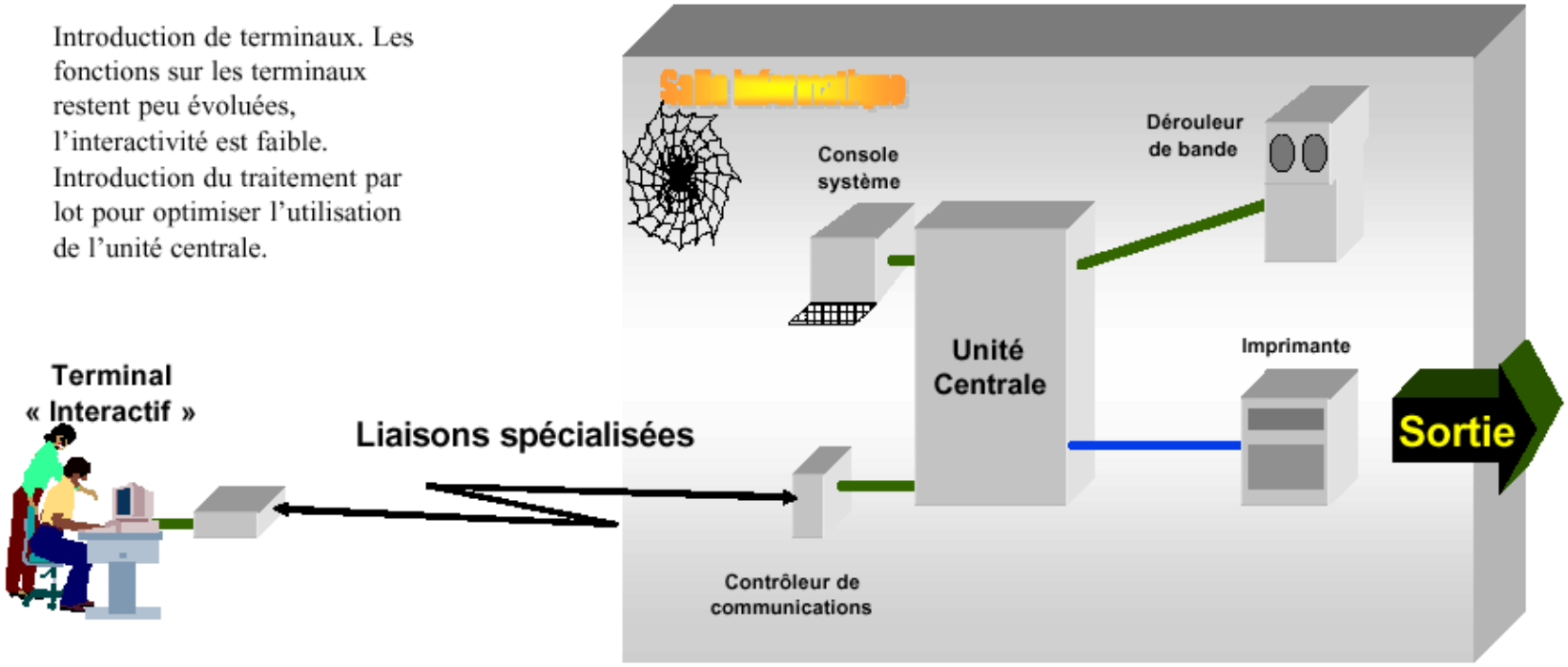


On peut difficilement parler d'interactivité: le programme ainsi que ses paramètres sont entrés dans l'unité centrale à l'aide de cartes perforées…
Une erreur sur une seule carte et tout le programme doit être relancé.
Le résultat est généralement une impression papier sur une imprimante à rouleau.
La console système sert uniquement à contrôler l'activité de l'unité centrale et à ordonnancer les tâches (on parle de lots de travaux).
La première amélioration sera l'introduction de lecteur de bande magnétique.

Les dinosaures…

2. années 1970 – architecture centralisée / terminaux

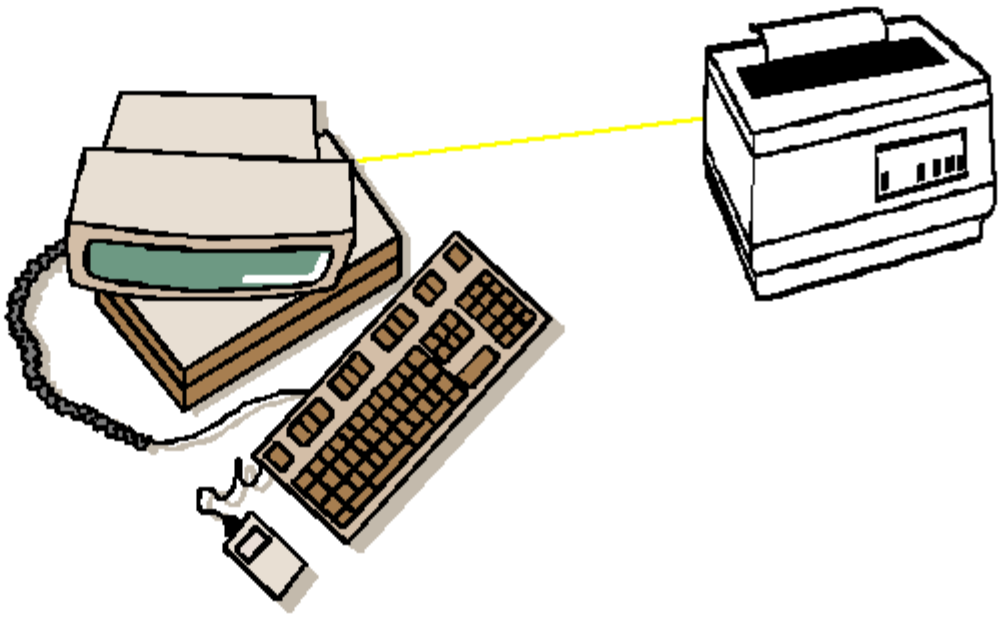
Introduction de terminaux. Les fonctions sur les terminaux restent peu évoluées, l'interactivité est faible. Introduction du traitement par lot pour optimiser l'utilisation de l'unité centrale.



La mutation….

3. Début des années 1980 – introduction de la micro-informatique

En 1981, IBM sort le premier Personal Computer.
C'est le début de la micro-informatique.



Les débuts difficiles de la distribution

3. Début des années 1980 (suite)– échange d'information par 'vélo processing'

L'échange d'information entre ordinateurs se fait par disquette ou autre média compatible (bande, …).

Ce moyen reste toutefois toujours parfois d'actualité lorsqu'il n'y a pas de connexion réseau, voire même pour des problèmes de sécurité (voir § sur la sécurité des réseaux).

On commence à se parler….

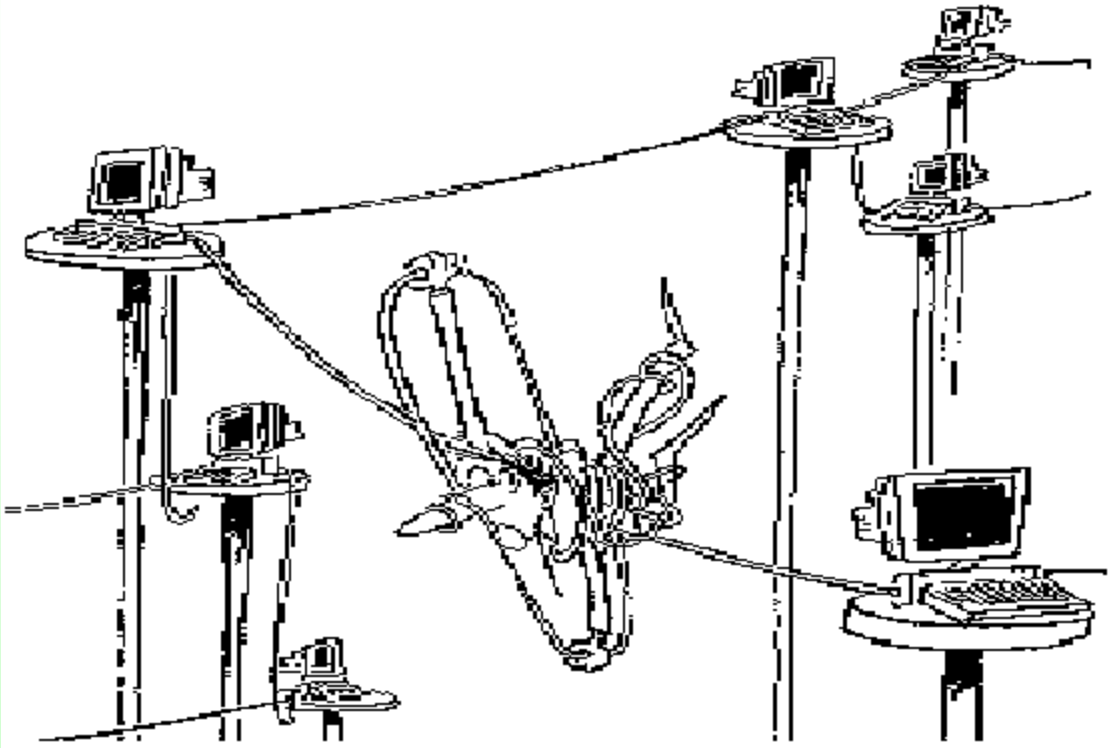
4. Fin des années 80 – Stations de travail & réseaux locaux

Les stations de travail prennent place pour les calculs scientifiques dans les universités et les entreprises.

Introduction des premiers réseaux locaux: Token-Ring, Ethernet.

L'infrastructure se met en place…

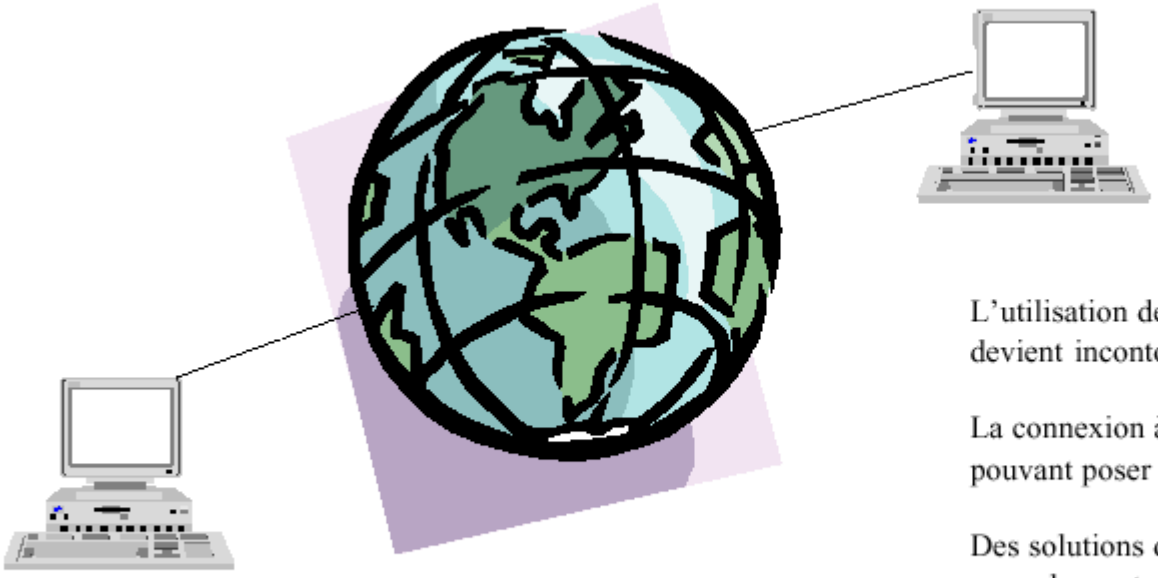
5. Milieu des années 90: Interconnexion des réseaux locaux



Au niveau des entreprises: l'interconnexion des réseaux locaux mis en place dans les différents services devient une nécessité.
L'architecture n'est pas toujours correctement spécifiée.

Les autoroutes de l'information….

6. Fin des années 90: Internet est incontournable.

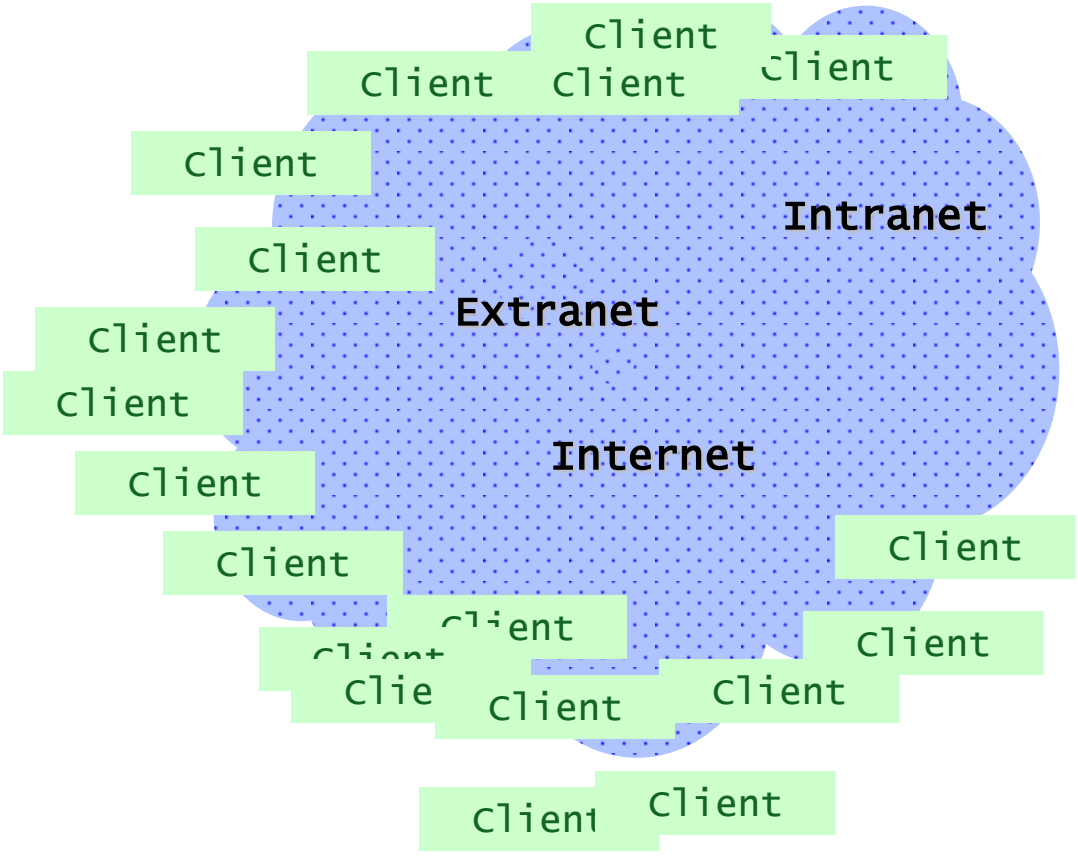


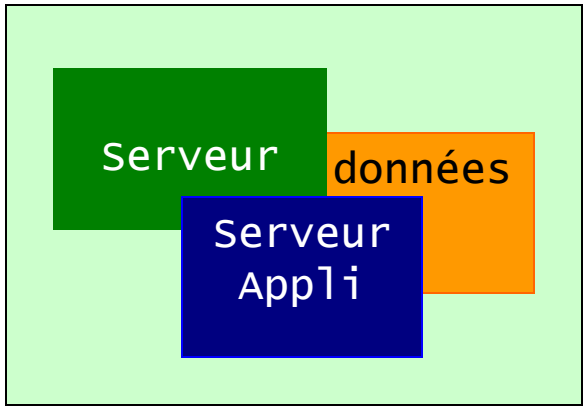
L'utilisation de technologie Internet devient incontournable.

La connexion à Internet d'une entreprise pouvant poser des problèmes de sécurité.

Des solutions de développement dans un vase clos restreint à l'entreprise, porte le nom d'Intranet ou encore d'Extranet, lorsqu'il s'agit de relier plusieurs points distants par des technologies Internet.

On commence à rationaliser….

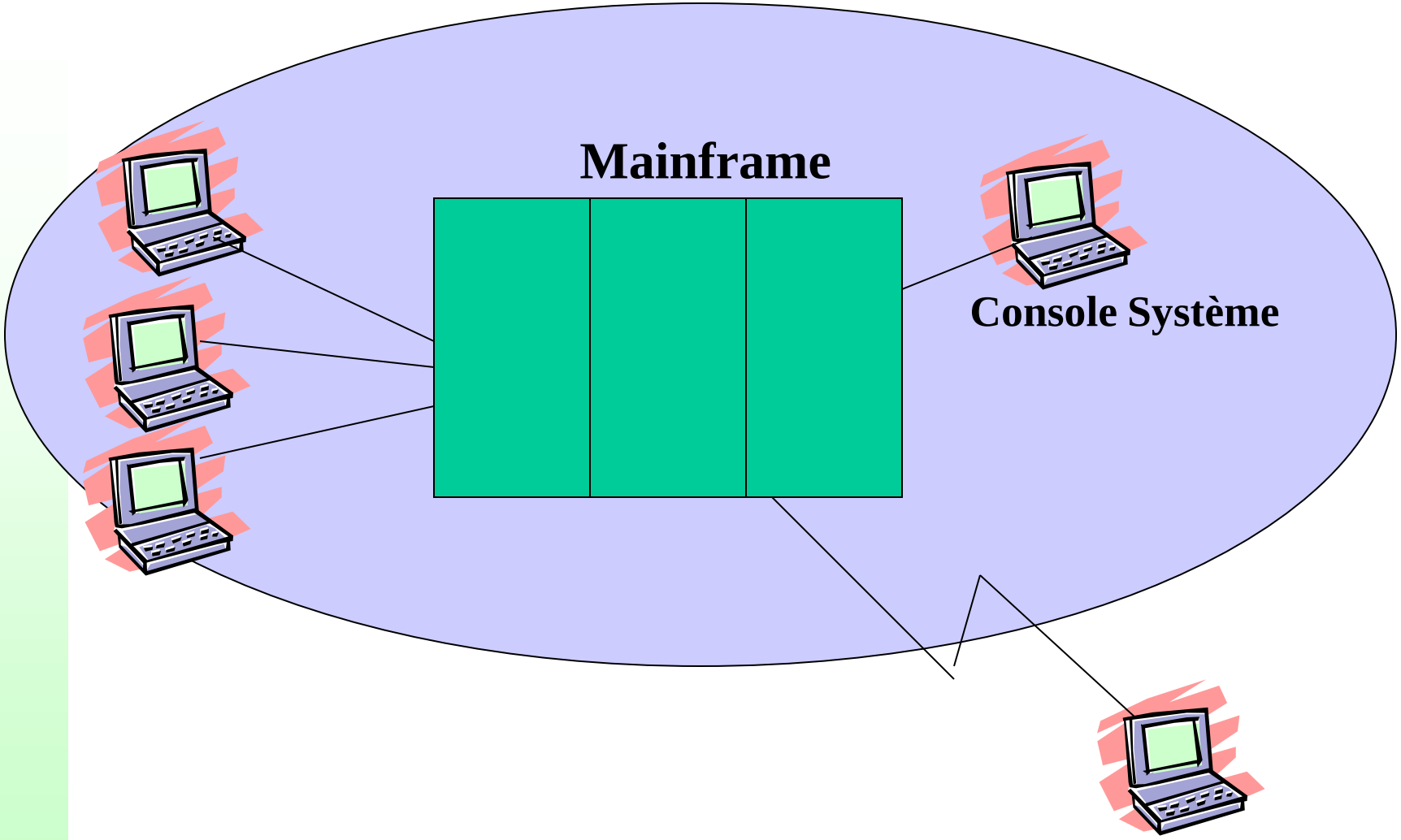




Le modèle Client Serveur

- Caractérisation des systèmes centralisés :
  - Tous les traitements s ’effectuent sur un seul système
  - Operating system généralement propriétaire
  - Terminaux en mode caractère (dumb terminal)
  - Exemples : IBM MVS, GCOS BULL, VMS DEC, Unix avec émulation de terminal

- Architecture Centralisée

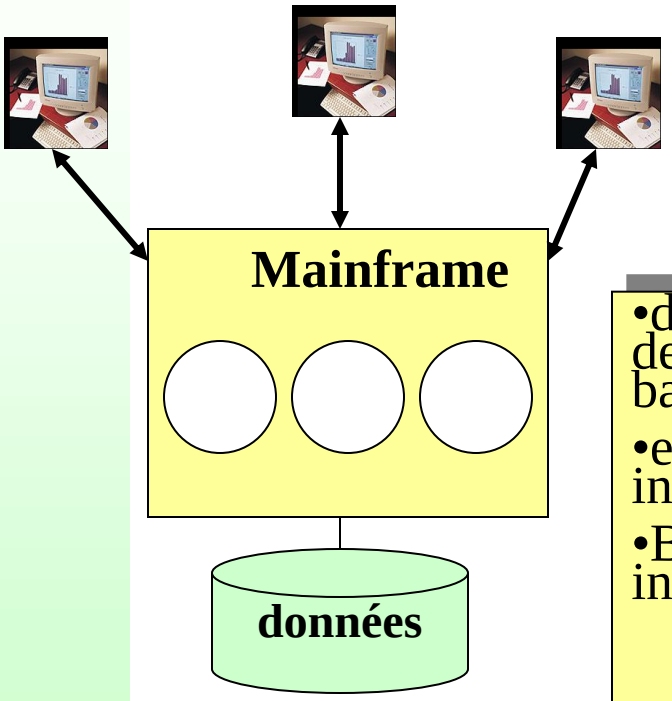


Avantages/inconvénients des Systèmes Centralisés

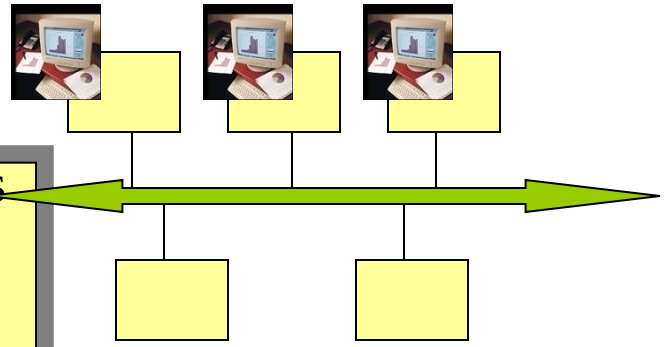
- Avantages :
  - Sécurité plus facile à gérer
  - Coûts d ’administration faible par utilisateur
  - Centralisation des choix:
    - Installation
    - Mise à jour
    - Utilisation des applications
- Inconvénients :
  - Interface utilisateur difficile à utiliser
  - Faible autonomie de l ’utilisateur
  - Systèmes propriétaires
  - Difficile de migrer vers un autre Système

Du modèle centralisé au client-serveur

Modèle centralisé



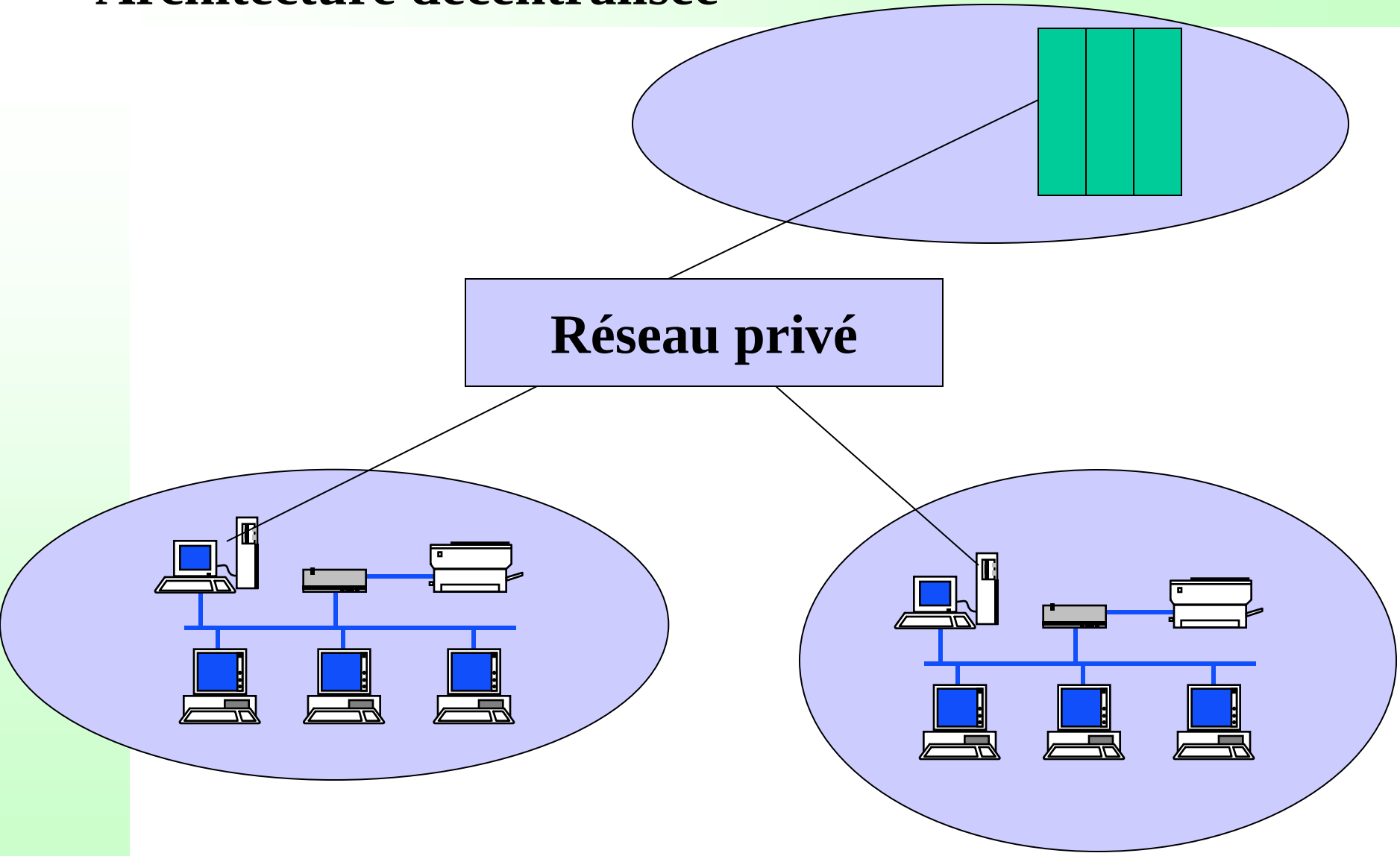
Modèle de l'informatique répartie



- développement des postes de travail banalisés à coût bas et des réseaux
- entreprise étendue, intégration de sites distants
- Besoin en applications interactives

- Caractérisation des systèmes décentralisés :
  - Les applications sont découpées en composants s ’exécutant sur des machines différentes
  - En général, une partie cliente et une partie serveur. Modèle Requête/réponse.
  - Downsizing : Décentralisation des traitements
  - Data Distribution : Répartition des données
  - Revamping : habillage d ’ancienne application et relookage graphique
  - Apparition du « MiddleWare »
- Exemples : RPC Microsoft, CTOS, Requêtes SQL

Architecture décentralisée



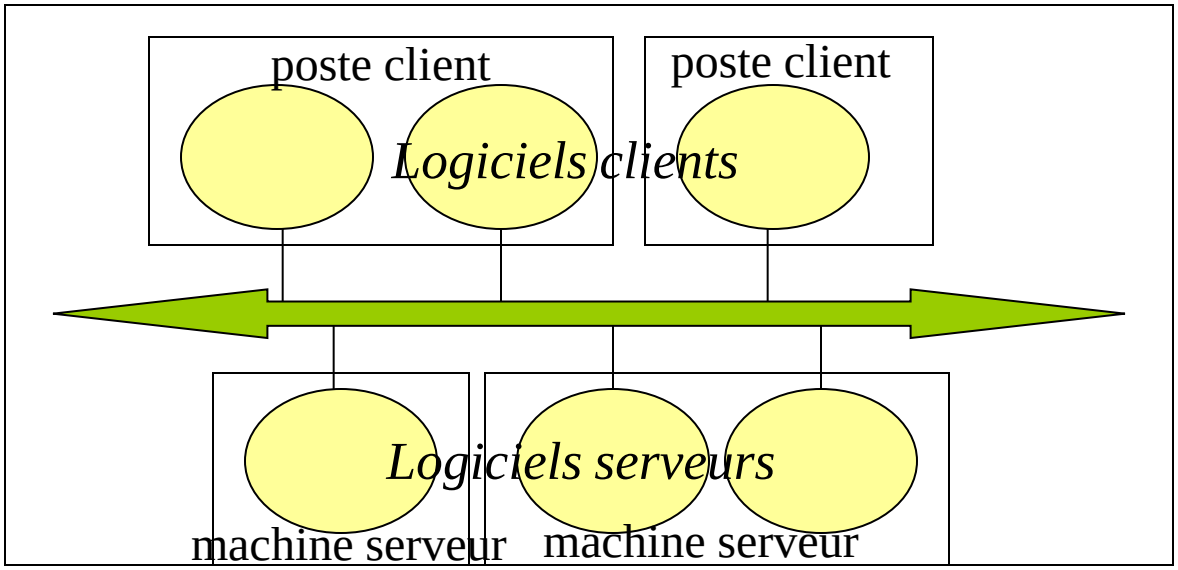
Avantages/inconvénients des Systèmes Décentralisés

- Avantages :
  - Répartition des traitements
  - Interface utilisateur plus conviviale (graphique)
  - Très grande autonomie de l ’utilisateur
  - Meilleur performance globale.
  - Données plus accessibles en cas de pbs réseaux
- Inconvénients :
  - Coût d ’administration plus élevé par utilisateur
  - Faible maîtrise des effets d ’échelle
  - Déploiement des applications plus compliquée
  - « fat client » et gestion des configurations clients plus problématique

3. Qu'est ce qu'une architecture distribuée?

- Une plate-forme pour construire des applications
- Définit les composants d 'un système et leur interaction
- Permet de construire un système complexe à partir d'éléments simples
- Permet de construire un système multi-parties qui se comporte comme un tout

Le « modèle » réparti



- Capacité d'évolution et d'adaptation
- distribution de la charge
- matériel banalisé et standard
- outils logiciels de coût réduit et de grande diffusion
- ouverture

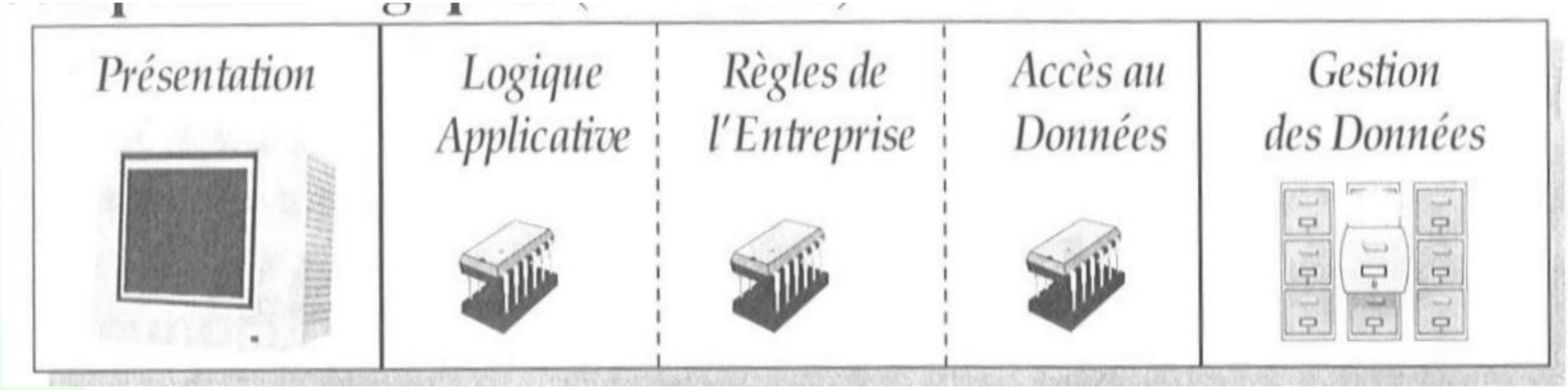
Mais :

- administration et déploiement
- construction de systèmes par intégration de progiciels

Familles technologiques de base :

- Systèmes de Gestion de Bases de Données
- Moniteurs transactionnels
- Systèmes de Groupware
- Objets distribués
- Serveurs d 'application
- Services Web

Les 5 composants logiques d ’une application



La stratégie d’affectation d ’un composant de l ’application à un système détermine son niveau de distribution

Problèmes liés à la distribution

- La complexité du fait de la multiplicité des composants
- L ’hétérogénéité
  - Multiplicité des langages
  - Multiplicité des plates-formes
- La distribution
  - Multiplicité des protocoles
  - Multiplicité des localisations possibles
  - La gestion du parallélisme et de la concurrence
- L ’intégration
  - Multiplicité des technologies
  - Multiplicités des interfaces

Le client serveur « simple », ou à deux niveaux

La séparation physique et logique
entre des clients et un serveur

Le client-serveur à 2 niveaux

Définition : approche d ’architecture qui vise à répartir un système informatisé sur des machines distantes, grâce à des matériels banalisés et des protocoles standards, et fondée sur une notion de service

Séparation d ’entités distinctes fonctionnant de concert pour accomplir une tâche :

– Composants logiciels clients
– Composants logiciels serveurs

⇨

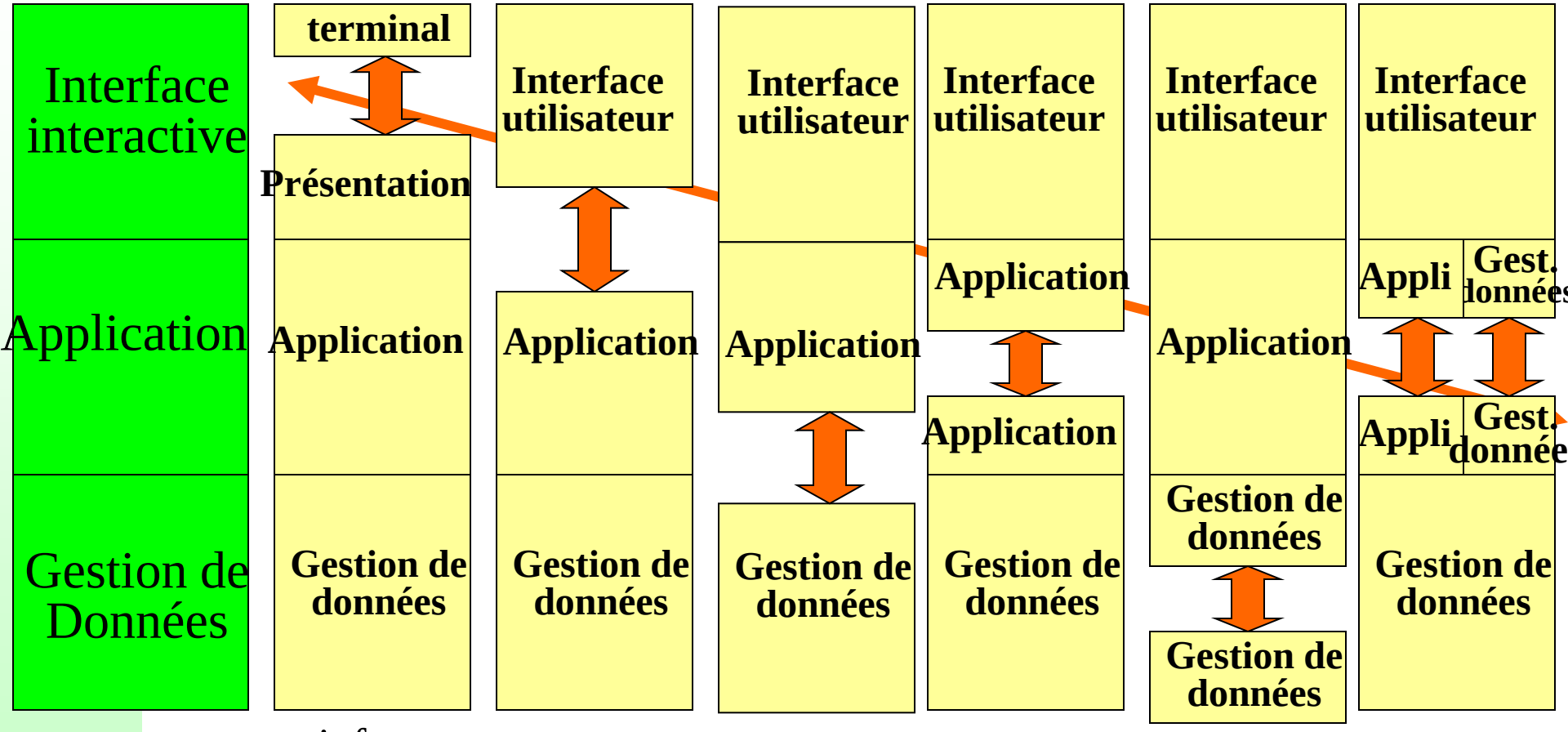
Problèmes :

- Définition d’une architecture adaptée à un contexte de besoins
- Où placer la « coupure » client-serveur ?

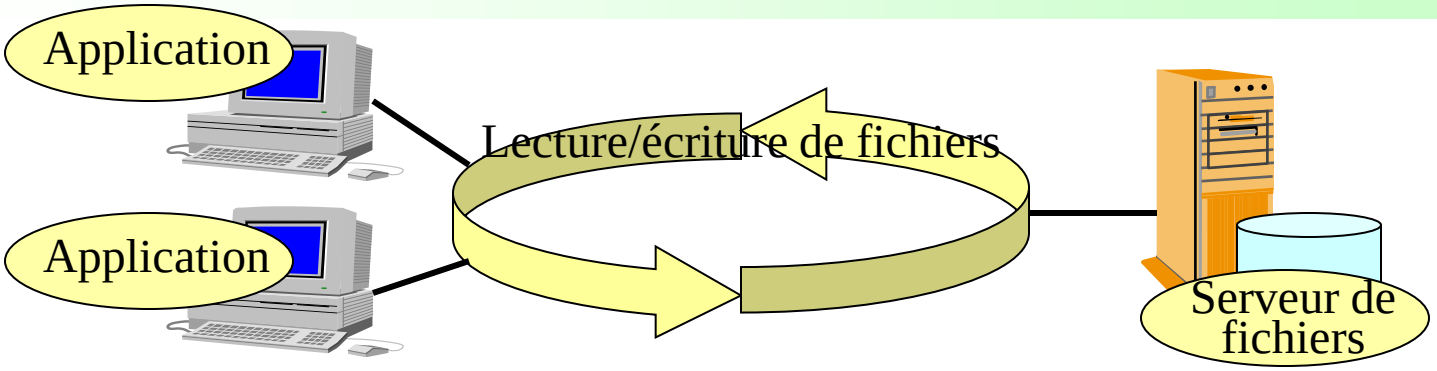
Caractéristiques du client-serveur

- Notion de service, avec « encapsulation »
- Partage de ressources
- Modèle d ’interaction de type requête
- Transparence relative à la localisation
- Indépendance vis à vis des matériels et des systèmes d ’exploitation
- Capacité d’évolution du système : ajout de stations clientes, changement de serveur, « passage à l’échelle »
- Intégrité des données partagées

Options d ’architecture

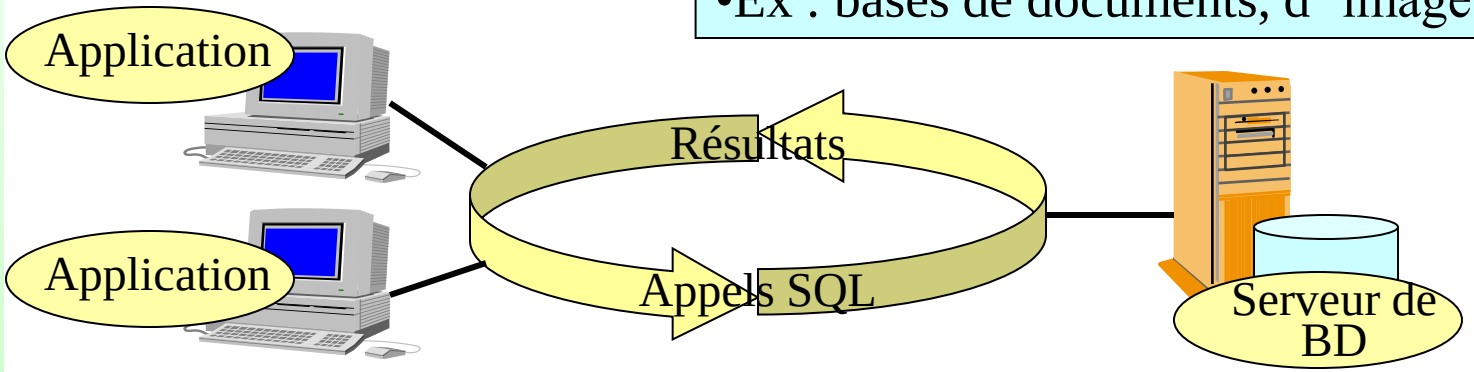


Communications client-serveur



Modèle Serveur de fichiers

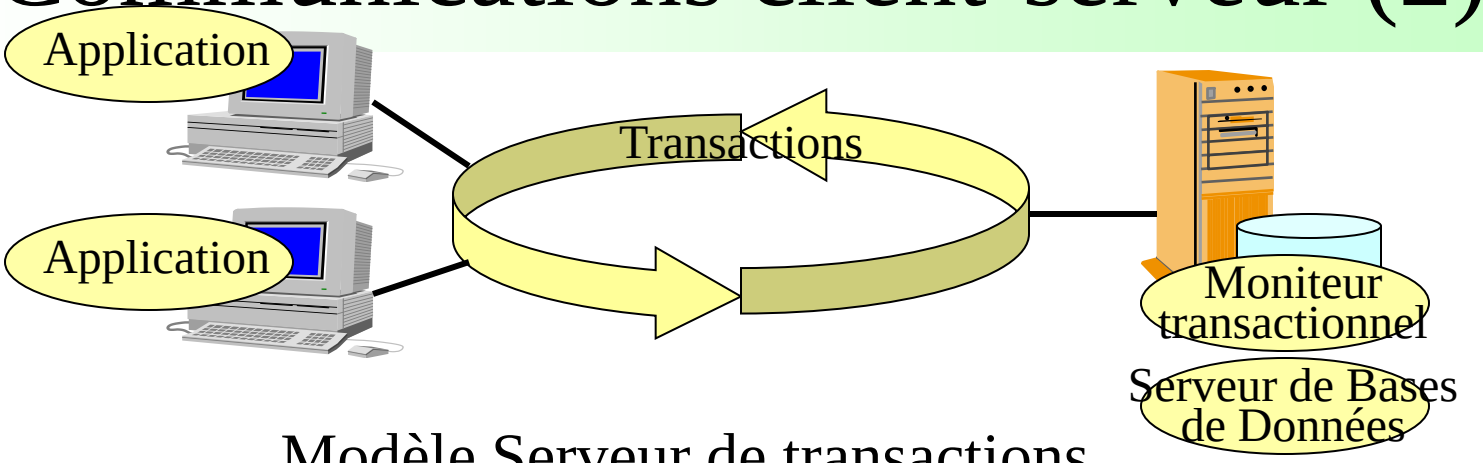
⇨
- Forme d ’échange de très bas niveau
- partage de fichiers sur un réseau (ex : NFS)
- Ex : bases de documents, d ’images, …



Modèle Serveur de bases de données relationnelles

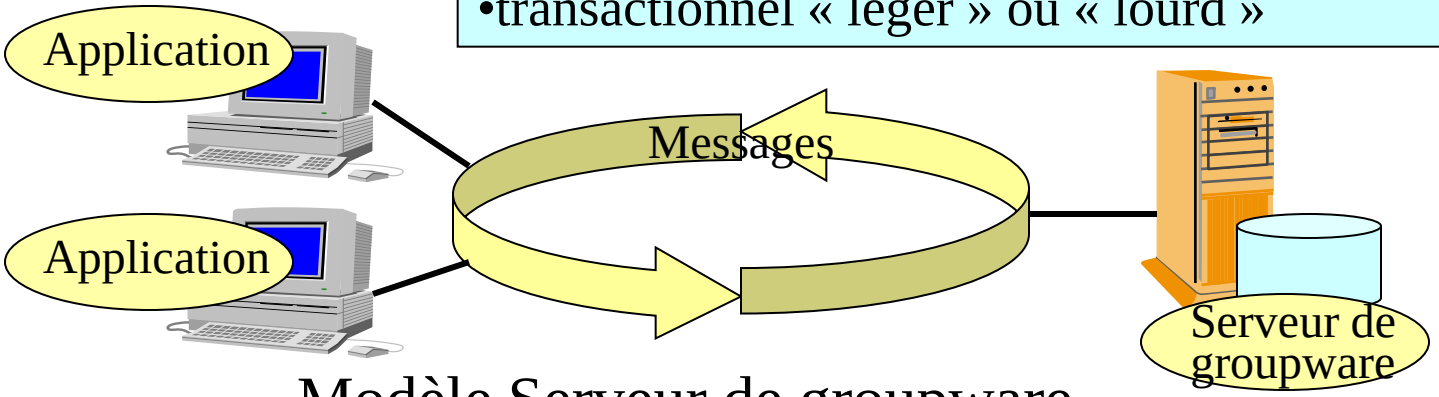
⇨
- Traitements de sélection sur le serveur
- utilisation du produit commercial d ’un éditeur

Communications client-serveur (2)



Modèle Serveur de transactions

- Un seul message pour un ensemble d ’opérations
- écriture de code sur client et serveur
- applications à temps de réponse critique (OLTP)
- transactionnel « léger » ou « lourd »



Modèle Serveur de groupware

- En général middleware propre à l ’éditeur
- tendance vers l ’utilisation de l ’email comme support aux échanges

Orientation client et orientation serveur

Partage de fonctions entre client et serveur

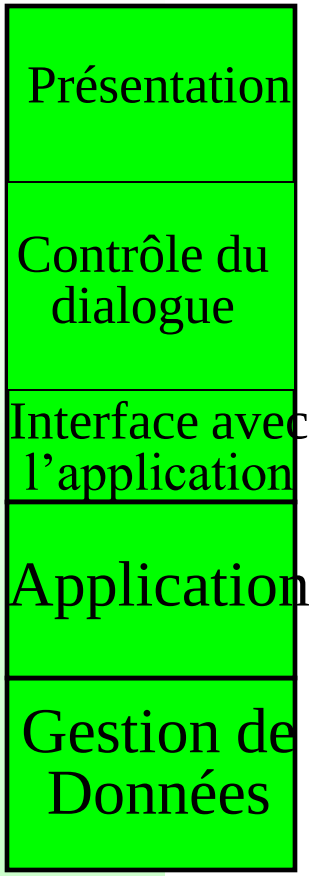
- orientation client. Ex : serveur de fichiers, SGBD
  logiciels individuels, aide à la décision
- orientation serveur. Ex : serveur de transactions
  déploiement et administration plus faciles
- approches combinées (ex : serveur de groupware)
- objets distribués

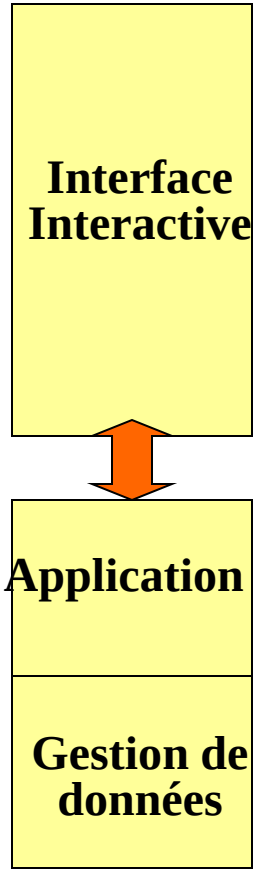
Client-serveur à deux niveaux

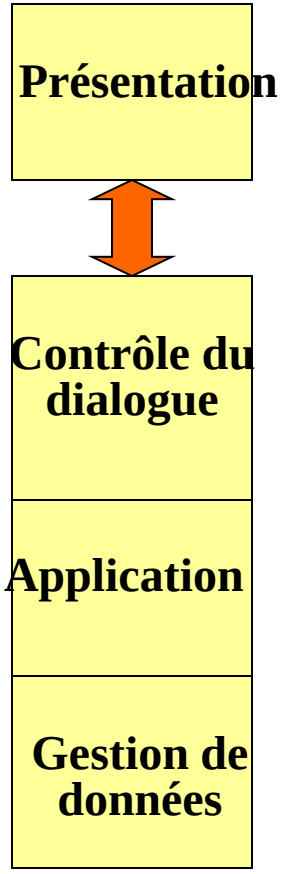
Logique applicative et traitements :

- dans le client avec l 'Interface Homme-Machine
- côté serveur ou dans la base de données
- répartie entre les deux

Extension à l'Interface Homme-Machine







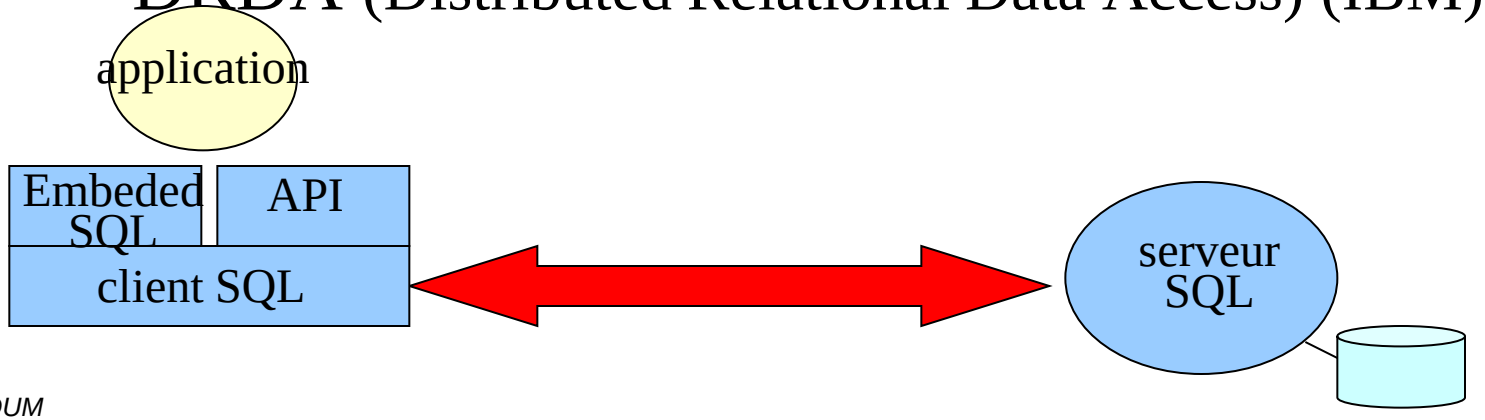
Ex : X11, Citrix, saisie de formulaires Web

Différents modes d'interaction entre clients et serveurs

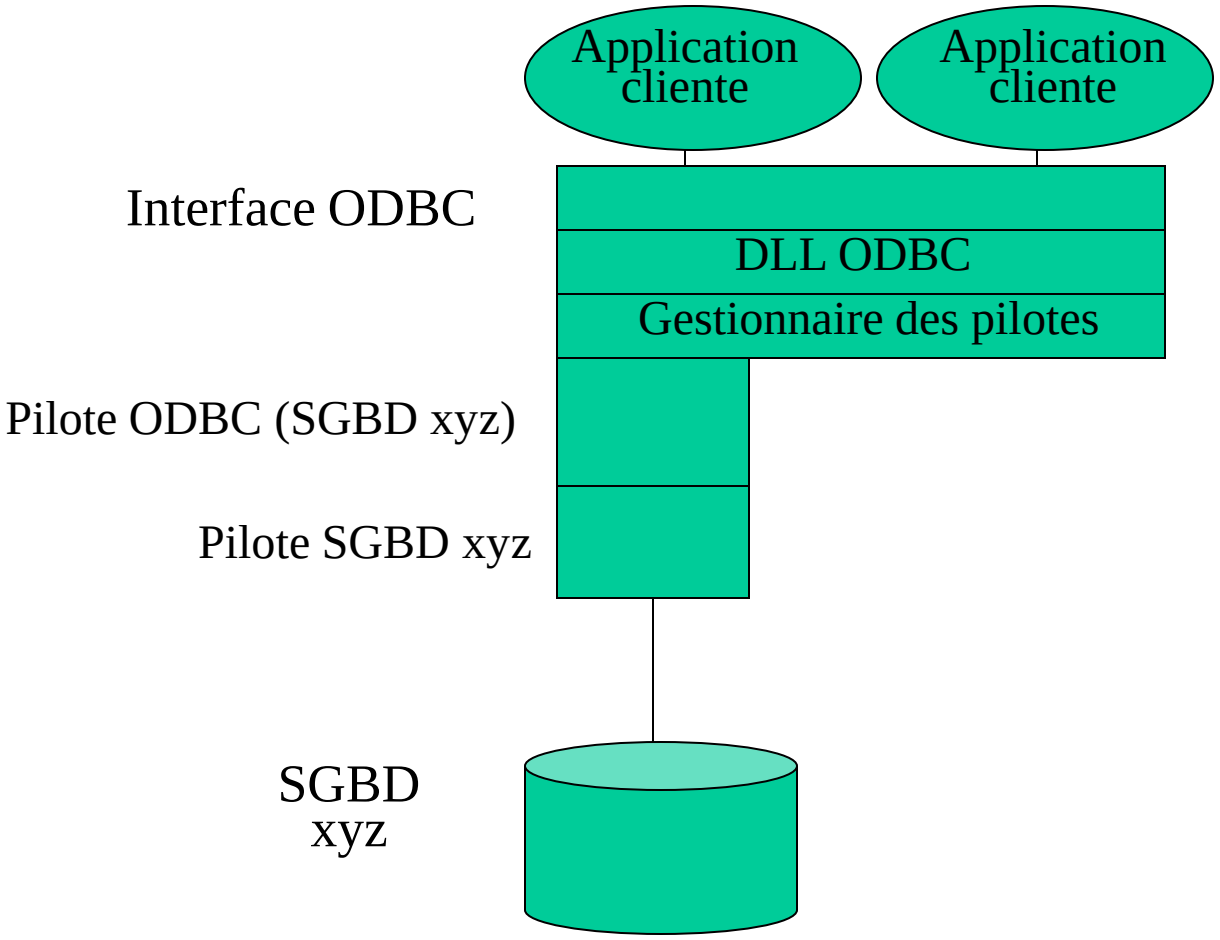
SGBDR, RPC, moniteur transactionnels….

Accès aux bases de données relationnelles distantes

- SQL standard SQL2, SQL3 + extensions
- Interfaces : des fonctions de base INSERT, UDPATE, SELECT, …
  - SQL intégré dans le code source (ESQL)
  - appel direct de SQL (API Call Level Interface) ⇨ ODBC (Open Data Base Connectivity) (SQL Access Group), JDBC (Java)
- passerelle ouverte : RDA (Remote Data Acces) (SAG), DRDA (Distributed Relational Data Access) (IBM)



Accès aux bases de données relationnelles distantes (2)



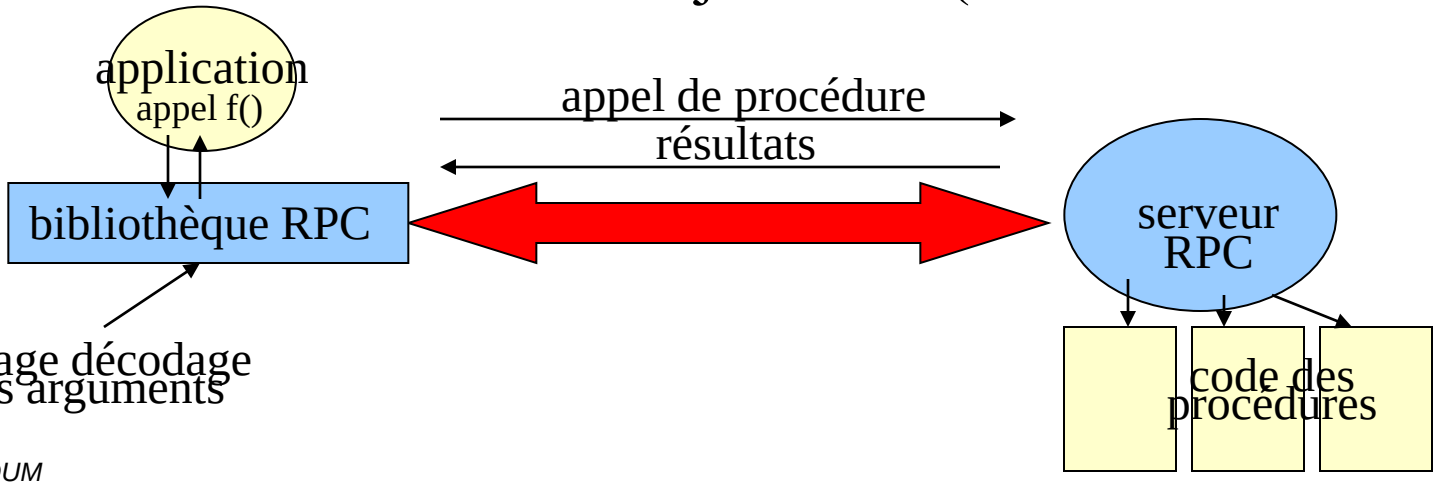
Source : « Serveurs multiprocesseurs, clusters et architectures parallèles ». René Chevance

Appel de procédures à distance

RPC (Remote Procedure Call)

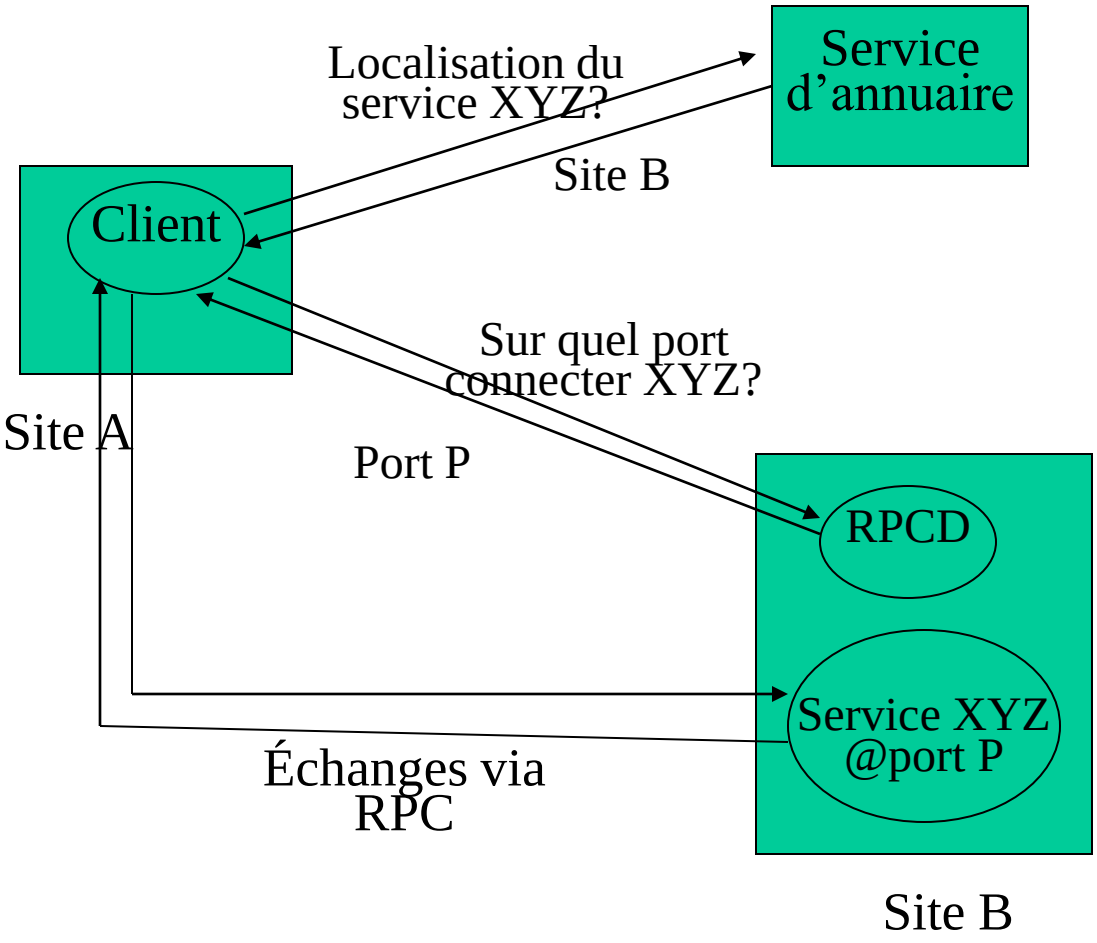
Principe : Appel d ’une procédure qui s ’exécute sur un site distant :

- invocation et attente de réponse (appel synchrone)
- quasi-transparence à l ’appel
- archivage de l ’entête (« signature ») des procédures (Interface Definition Language)
- évolution vers l ’objet : RMI (Remote Method Invocation)



Appel de procédures à distance (2)

Fonctionnement d ’un appel RPC



Source : « Serveurs multiprocesseurs, clusters et architectures parallèles ». René Chevance

Moniteurs transactionnels

Deux types d ’applications :

- Applications *transactionnelles*
- Application *décisionnelles*

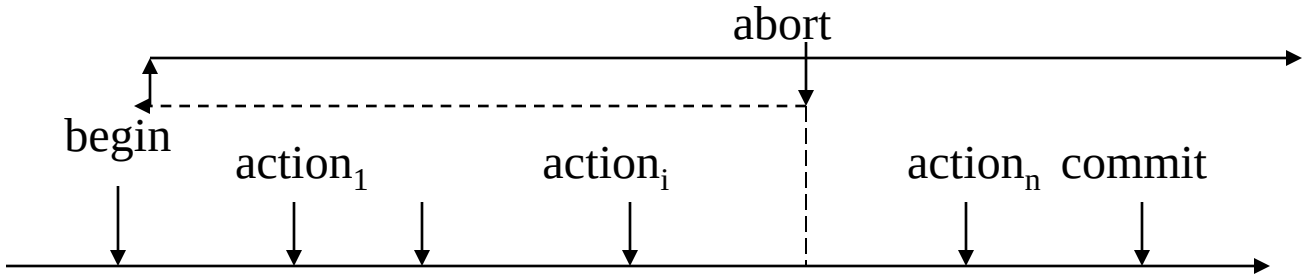
Applications transactionnelles : caractéristiques :

- bases de données partagées par l ’ensemble des utilisateurs
- flux de requêtes important et non planifié à l’avance
- travail répétitif (ensemble prédéfini de fonctions simples)
- grand nombre d ’utilisateurs (humains ou programmes)
- haute disponibilité
- intégrité des données fondées sur les propriété ACID
- Nécessité d’équilibrage de charge automatique

Propriétés ACID

Exécution d ’une séquence d ’actions

- **Atomicité** : les opérations sont exécutées dans leur totalité ou pas du tout (begin, commit, abort)
- **Consistance** : transformation correcte de l ’état du système (ne violant aucune contrainte d ’intégrité)
- **Isolation** : les changements sur les données ne sont visibles par les autres transactions qu ’une fois la transaction achevée
- **Durabilité** : les modifications qui suivent une validation doivent survivre à une panne du système



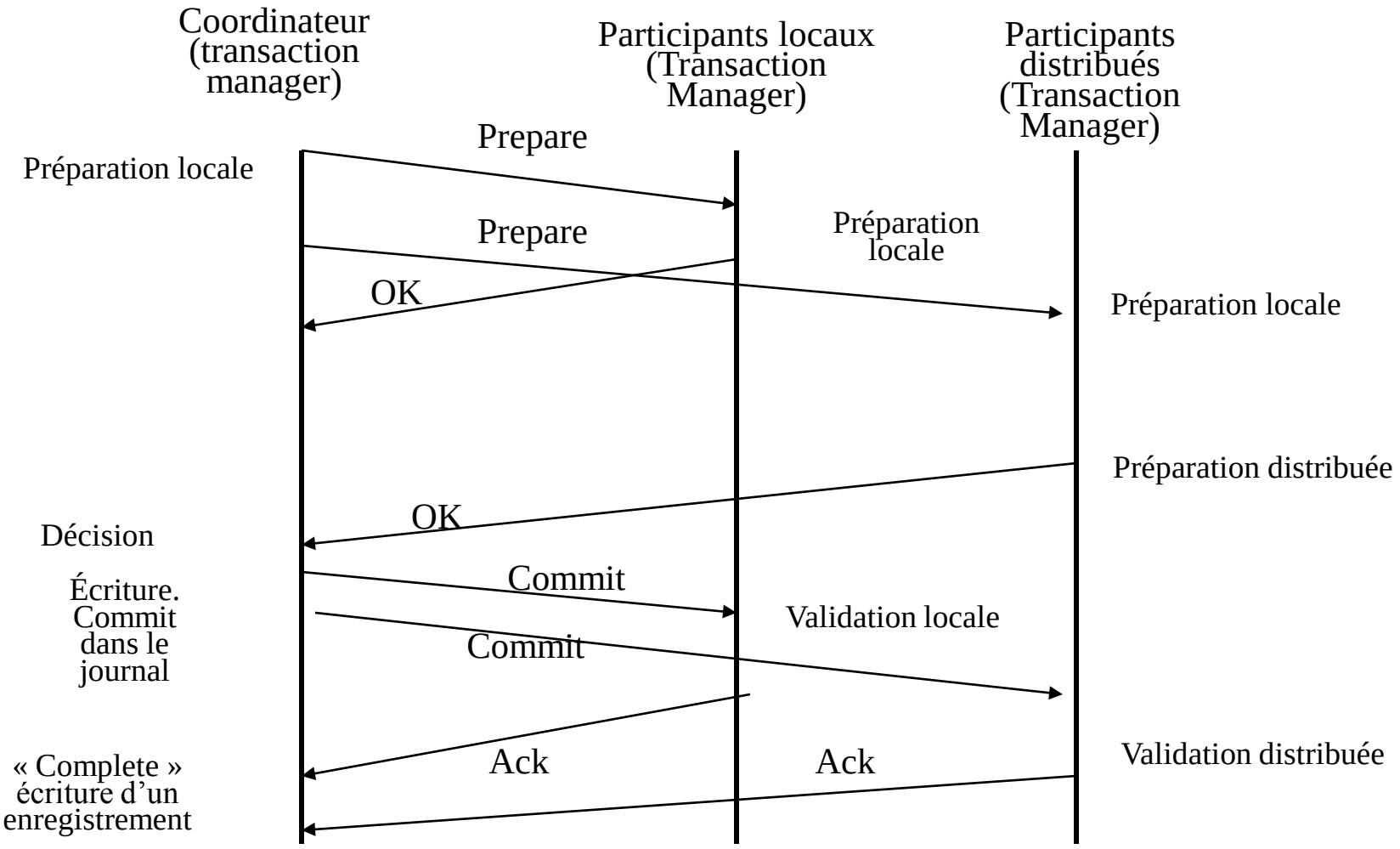
Fonctions d ’un moniteur transactionnel

Fonctions principales d ’un moniteur transactionnel :

- gestion des processus (équilibrage de charge, multiplexage des clients sur un nombre limité de processus)
- gestion des transactions (propriétés ACID) en contexte distribué (plusieurs gestionnaires de données)

- implémentation du protocole de validation à deux phases (two phases commit) en coopération avec les gestionnaires de données
- modèle d ’architecture défini par X/Open (Open Group) : interfaces TX, XA, XA+

Principe de la validation à 2 phases



Source : « Serveurs multiprocesseurs, clusters et architectures parallèles ». René Chevance

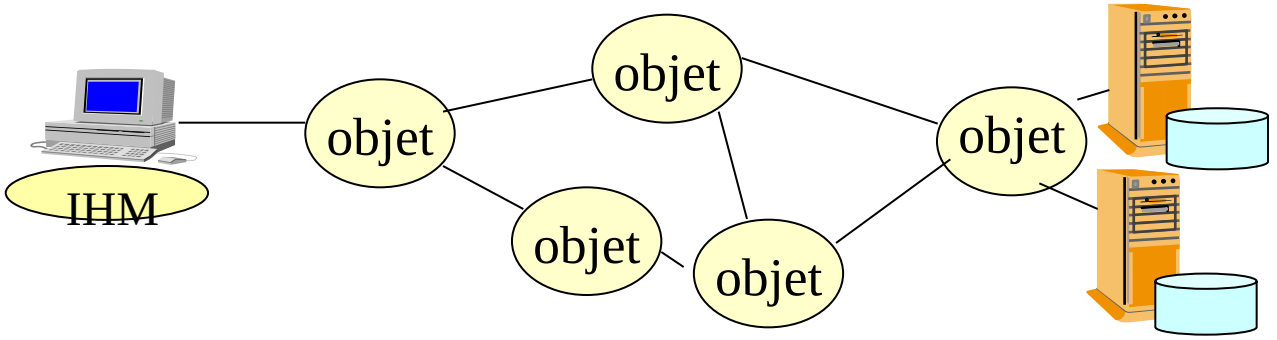
Une première évolution : le modèle à objets distribués

Répartition des traitements et des données sur un réseau de machine

Modèle à objets distribués

Coopération de services distribués vus comme des objets

Un objet peut être alternativement client et serveur

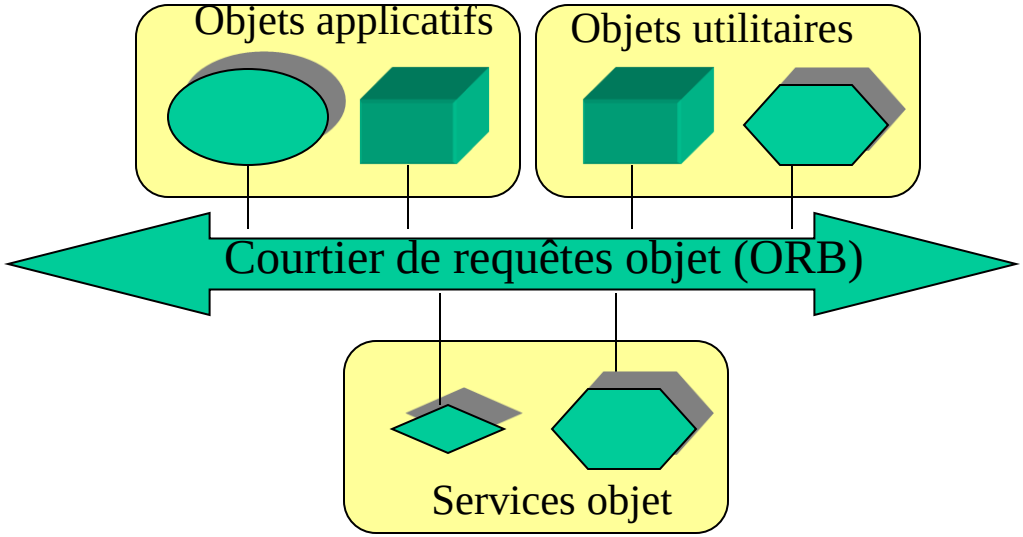


Deux modèles :

- CORBA (Common Object Request Broket Architecture) de l ’OMG
- COM+ (Microsoft)

Modèle à objets distribués

Modèle de référence CORBA

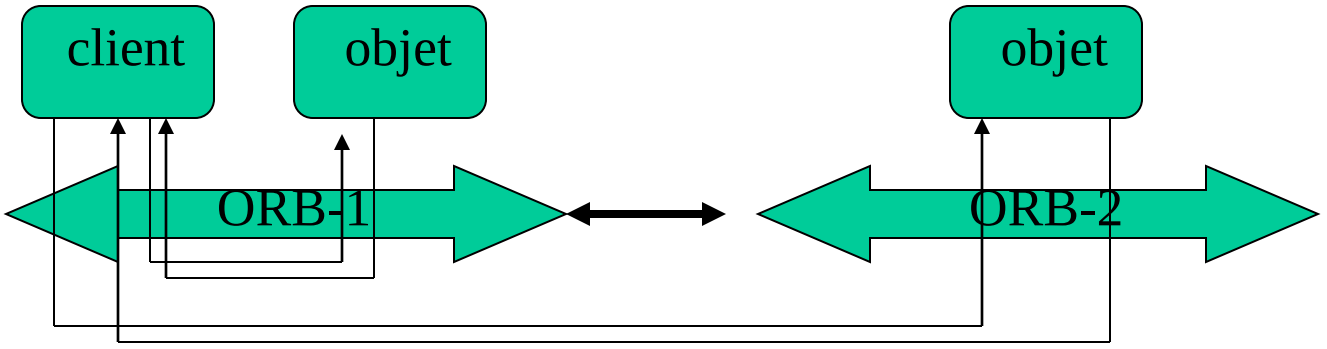


Source : « Serveurs multiprocesseurs, clusters et architectures parallèles ». René Chevance

Éléments du modèle :

- Objets applicatifs
- Objets utilitaires : services d ’emploi général (catalogues de classes, messagerie, …)
- Services objet utilisables dans certains environnements (création d ’instances, services transactionnels, persistance, nommage, …)
- Courtier d ’objets (Object Broker)

Échanges avec CORBA

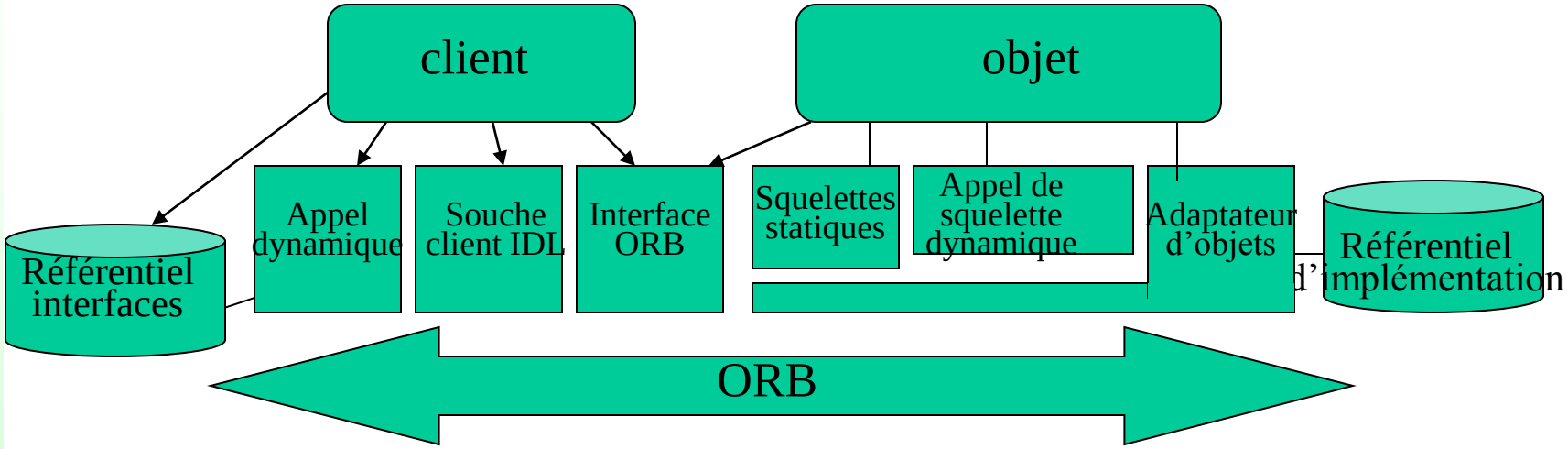


Source : « Serveurs multiprocesseurs, clusters et architectures parallèles ». René Chevance

Échanges :

- entre objets liés par un même ORB
- entre ORB de différentes implémentations (protocole IIOP : Internet Inter-ORB Protocol)

Architecture fonctionnelle de CORBA



Source : « Serveurs multiprocesseurs, clusters et architectures parallèles ». René Chevance

Les interface Web et le « client léger »

L’utilisation des technologies Internet

Évolution liées à Internet

Internet

- Interconnexion de réseaux à l ’échelle mondiale fondée sur les protocoles TCP/IP
- Applications : Web, email, ftp, .. avec protocoles associés

Tendances actuelles

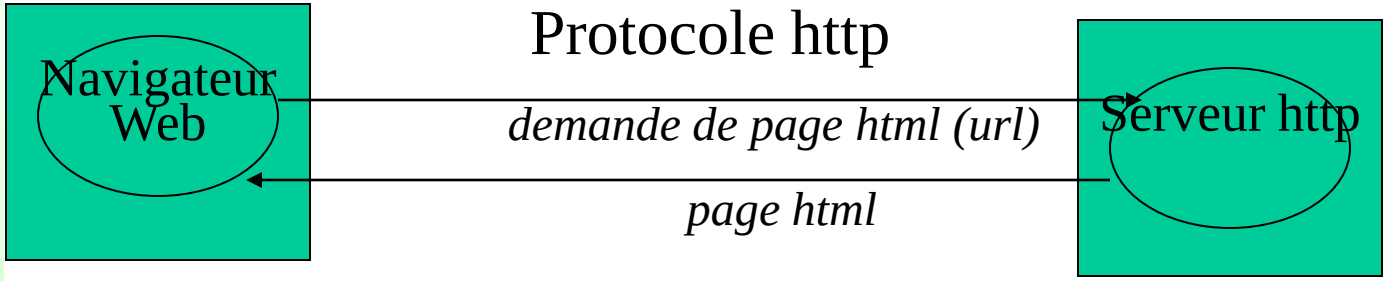
- Utilisation dans l ’entreprise d ’architectures de type Intranet / Extranet
- Montée en charge des applications liées au e-commerce B to B, B to C, ...

⇨

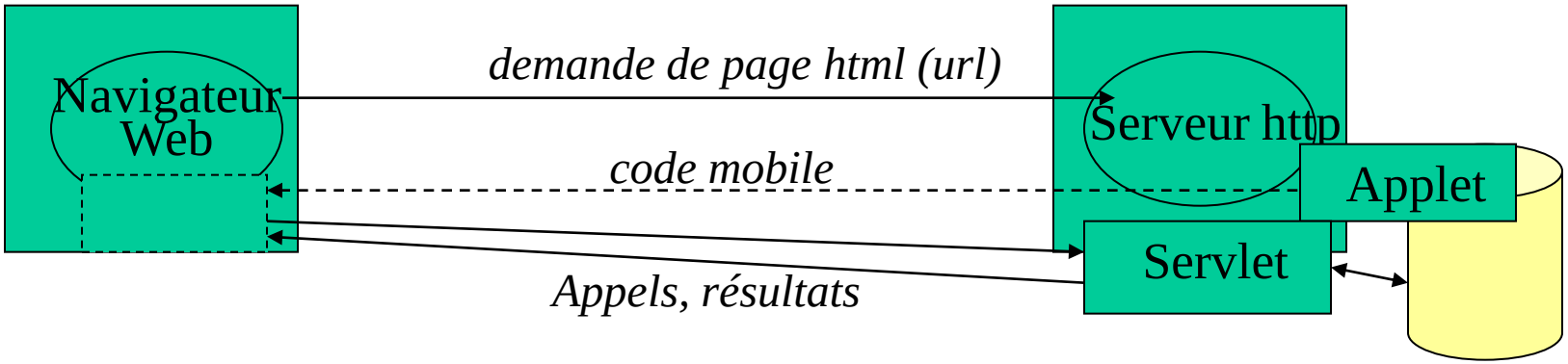
- Évolution client-serveur sur réseau local ⇨ Client-serveur sur Internet
- Modèle du « client léger », mode « non connecté »
- Exploitation des « technologies Internet » dans le modèle client-serveur

Client léger et technologie Web

Problème du déploiement d'applications du un grand réseau ⇨ temps, coût, incohérences entre versions



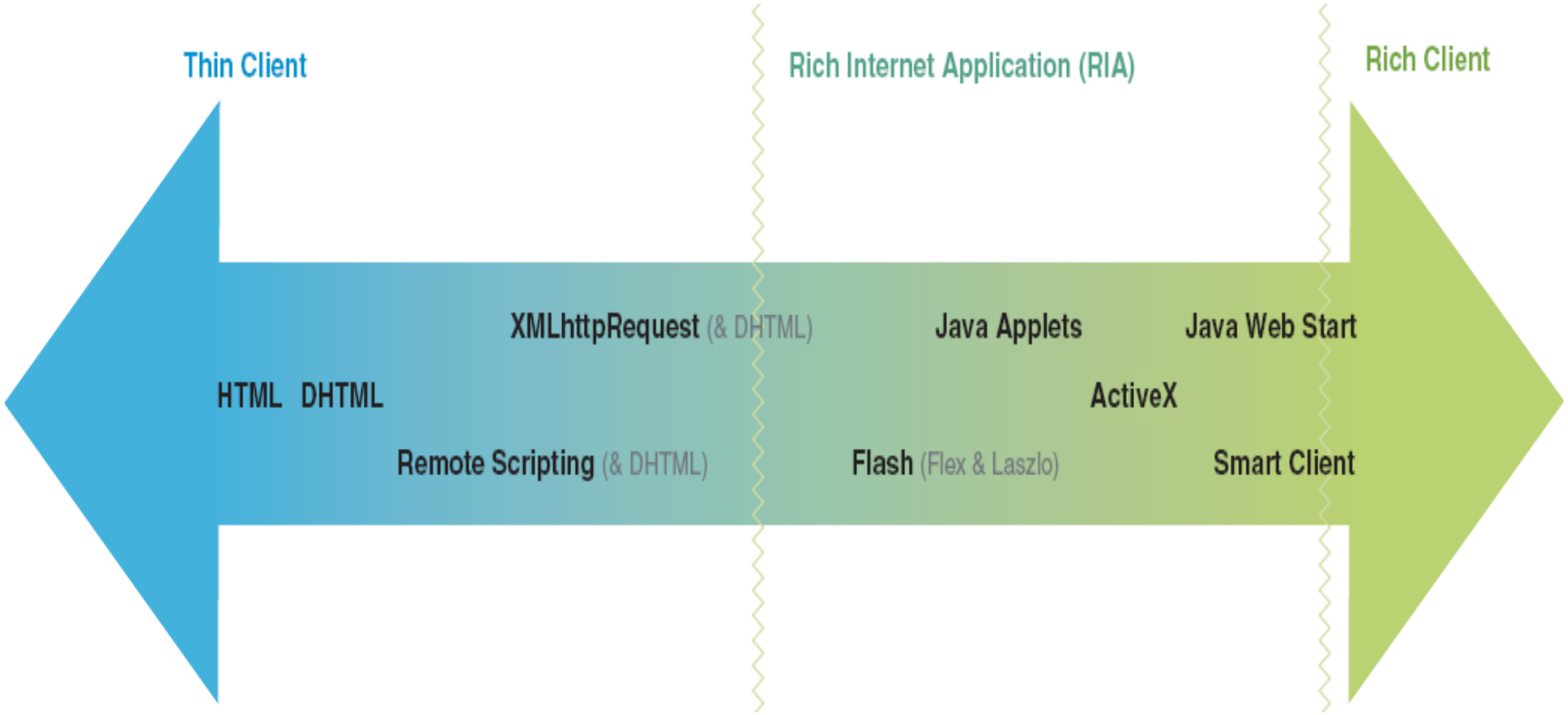
Client riche et technologie WEB



- Besoin d'améliorer l'interface utilisateur
- Asynchronisme entre les échanges avec le serveur et les saisies utilisateur

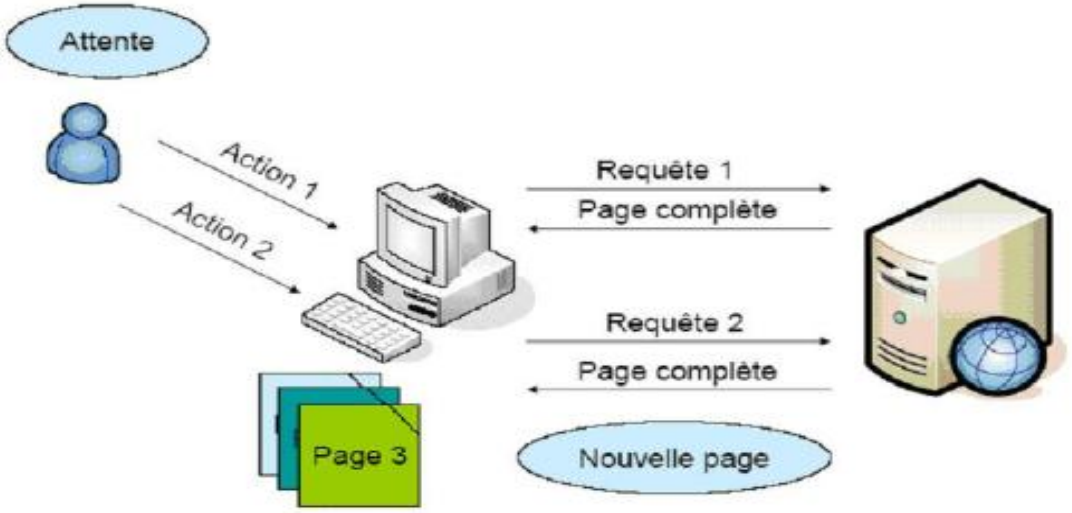
Client riche et technologie WEB

Client léger, RIA, client riche

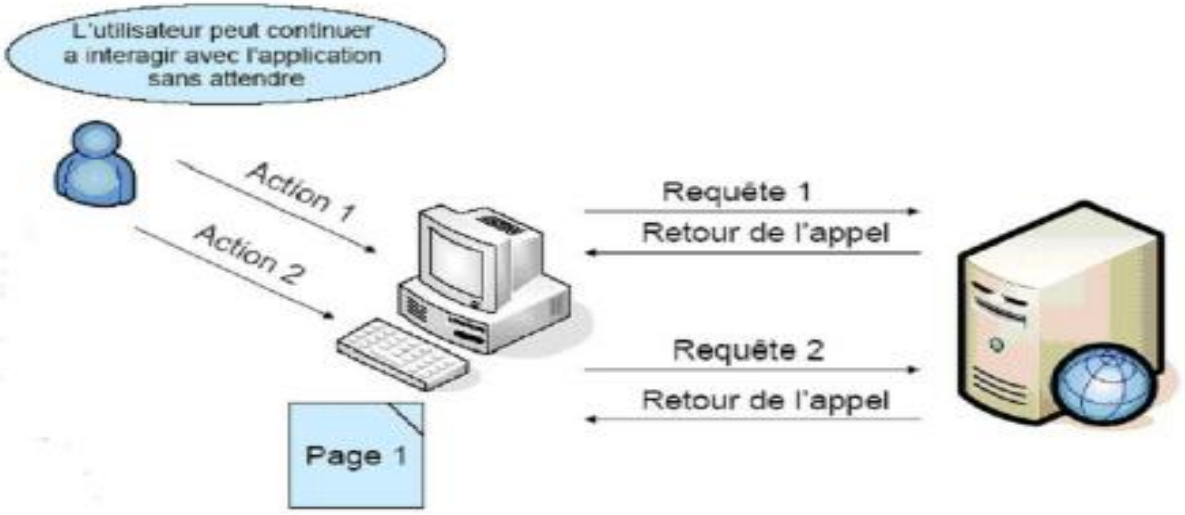


Différence de fonctionnement

Echange de données dans une page web traditionnelle



Echange de données dans une page web Ajax



PHP SCHEMA DE PRINCIPES

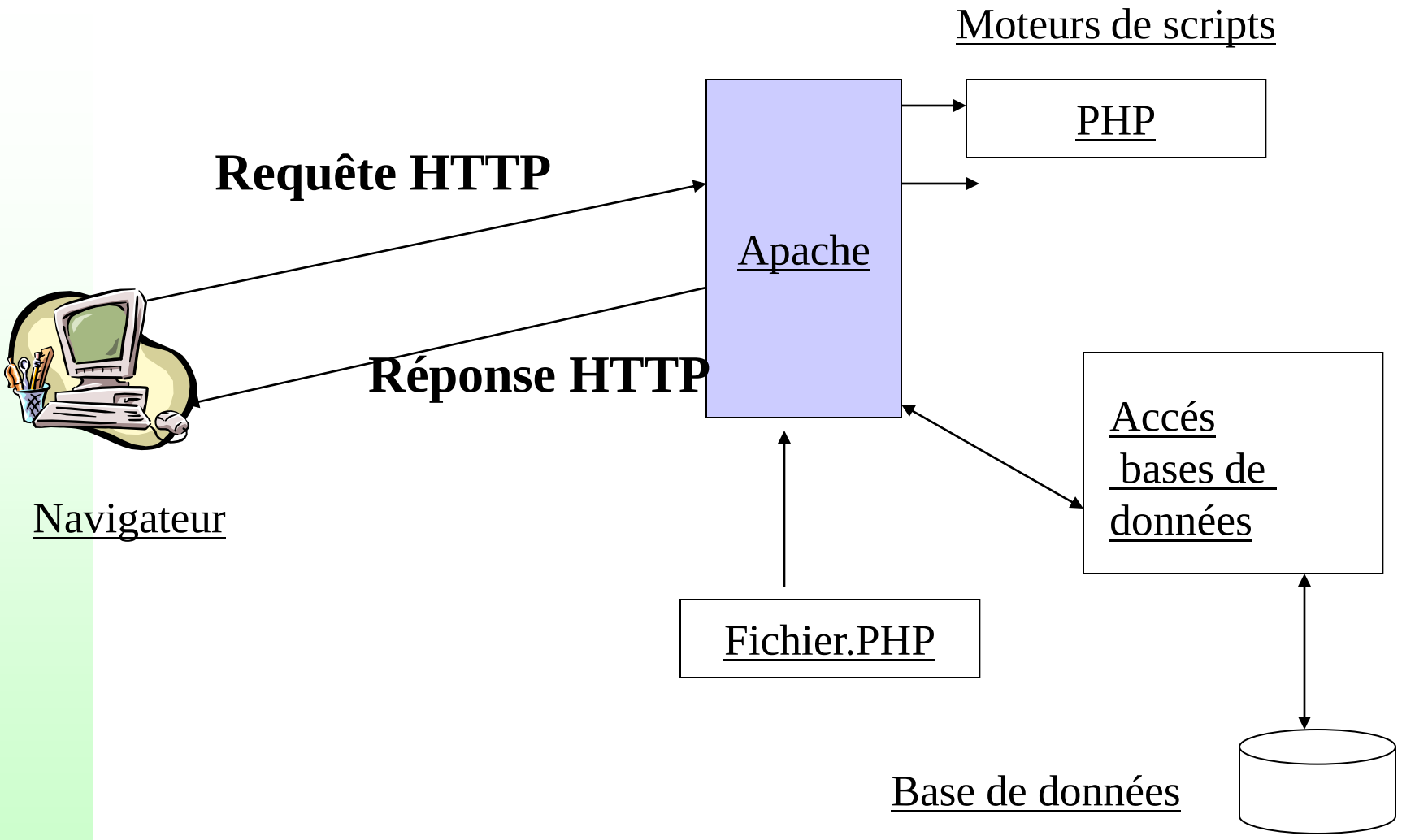


Schéma en ASP

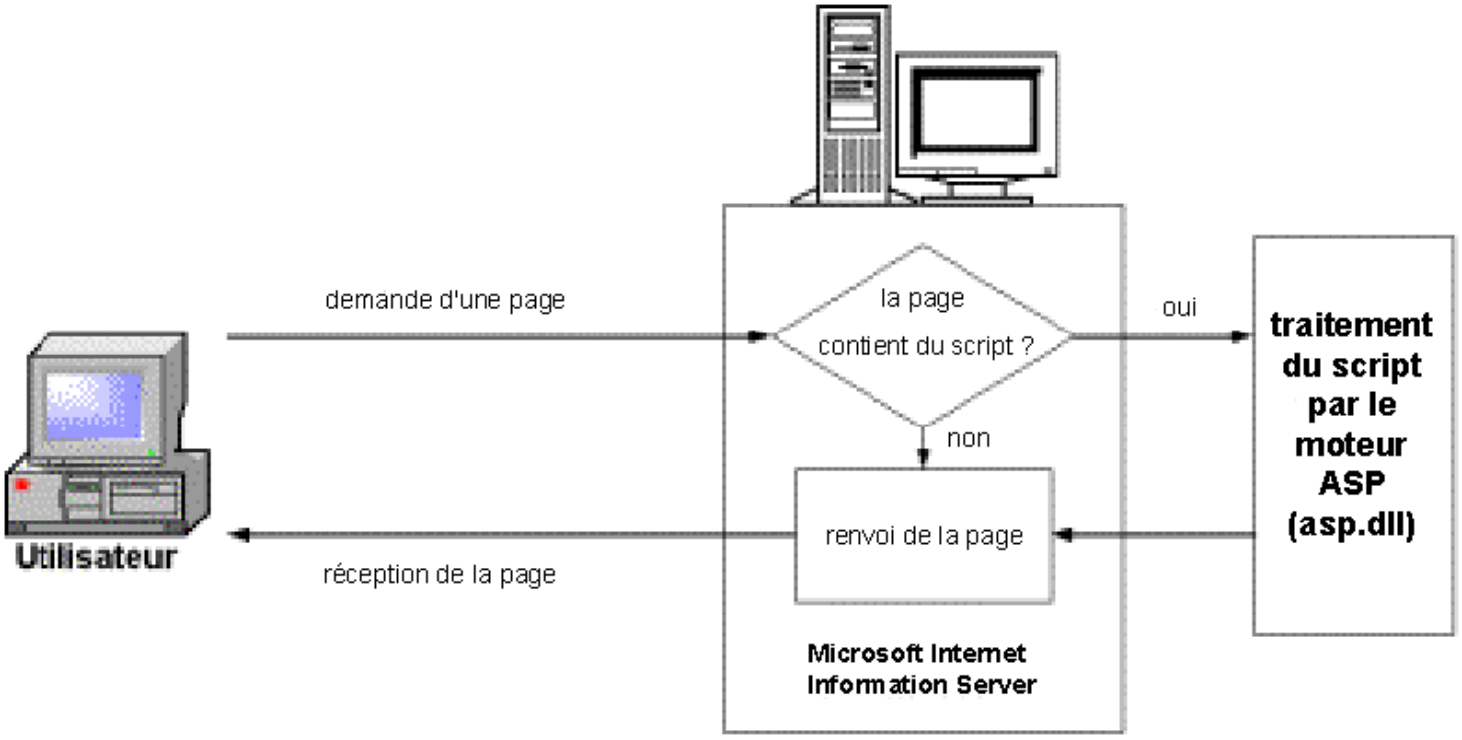
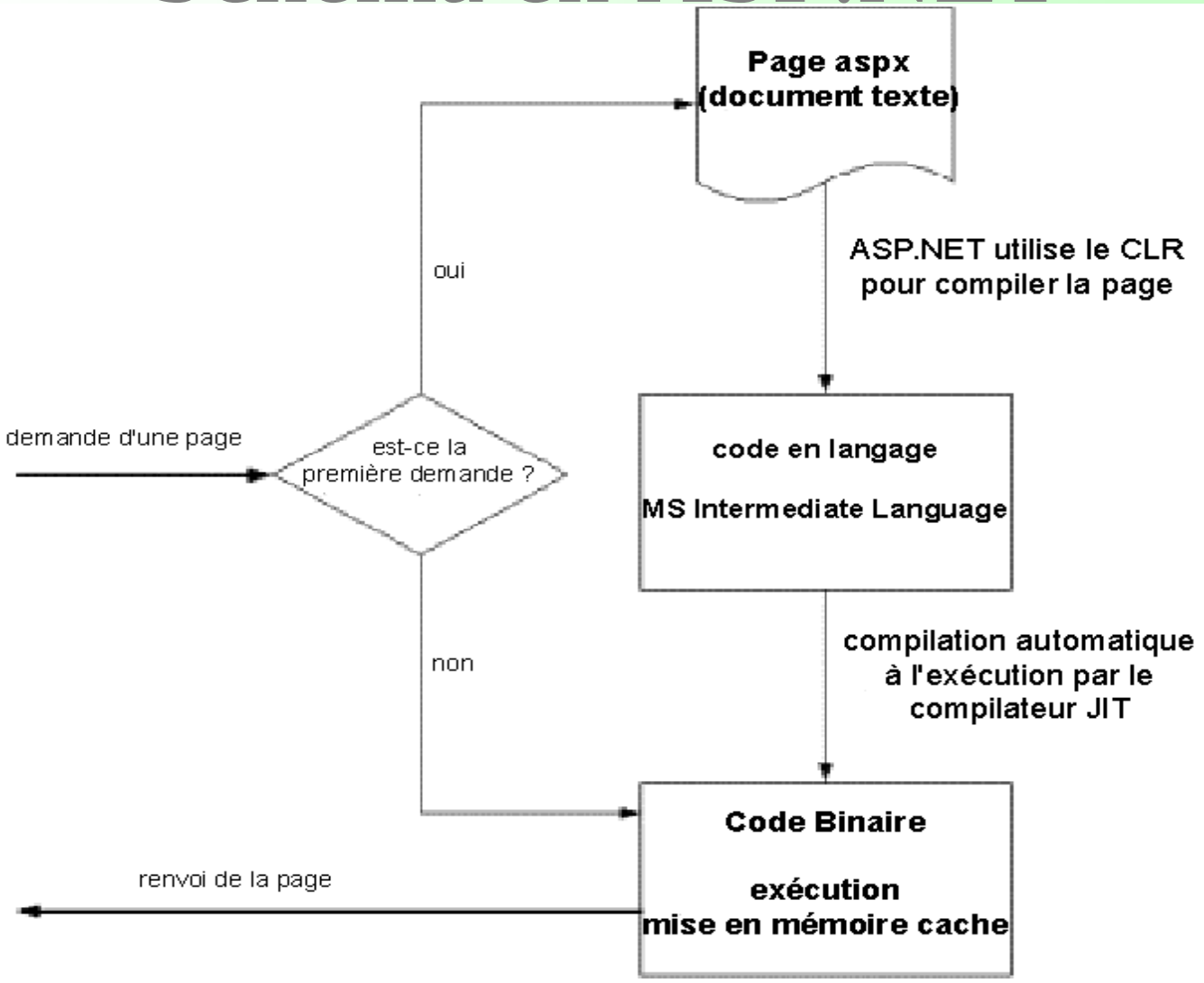


Schéma en ASP.NET

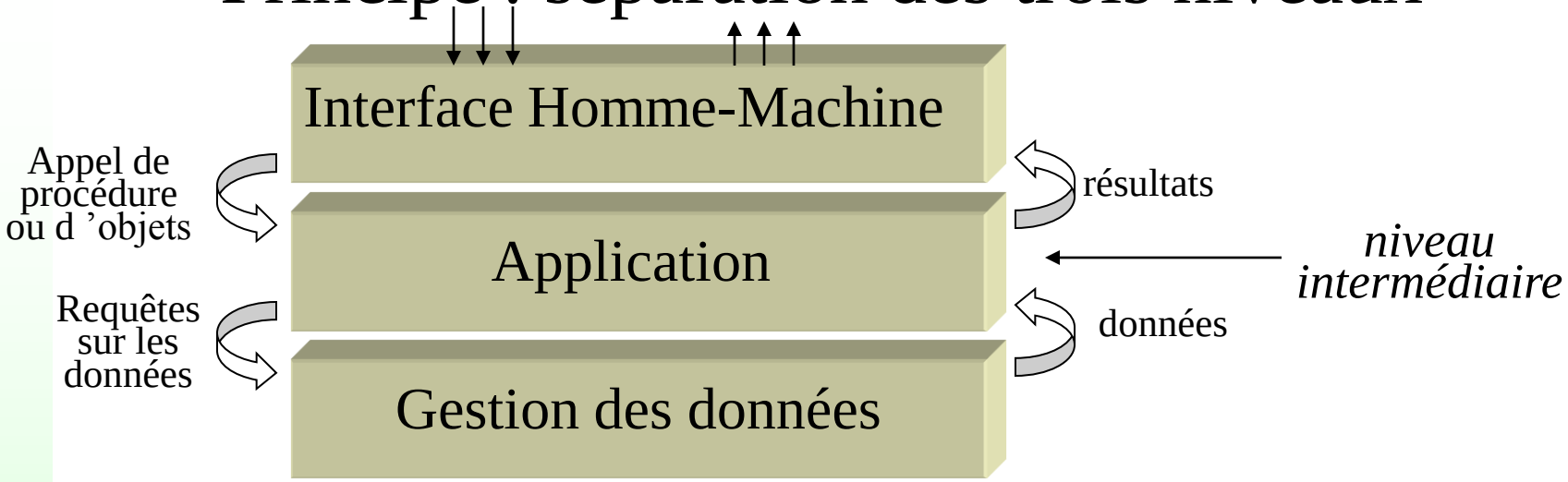


Les architectures 3-tiers et n-tiers

La structuration d'un application sur n niveaux

Architecture à trois niveaux

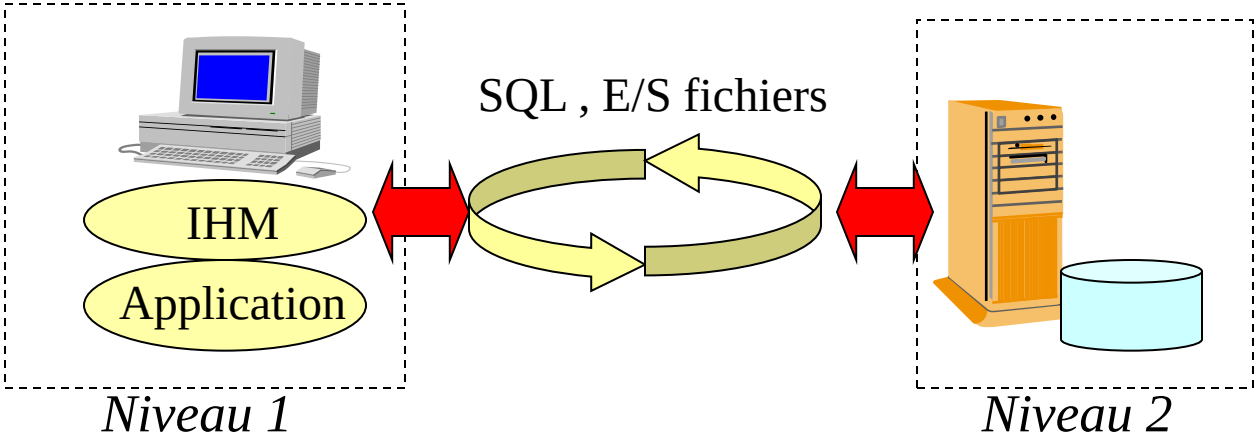
Principe : séparation des trois niveaux



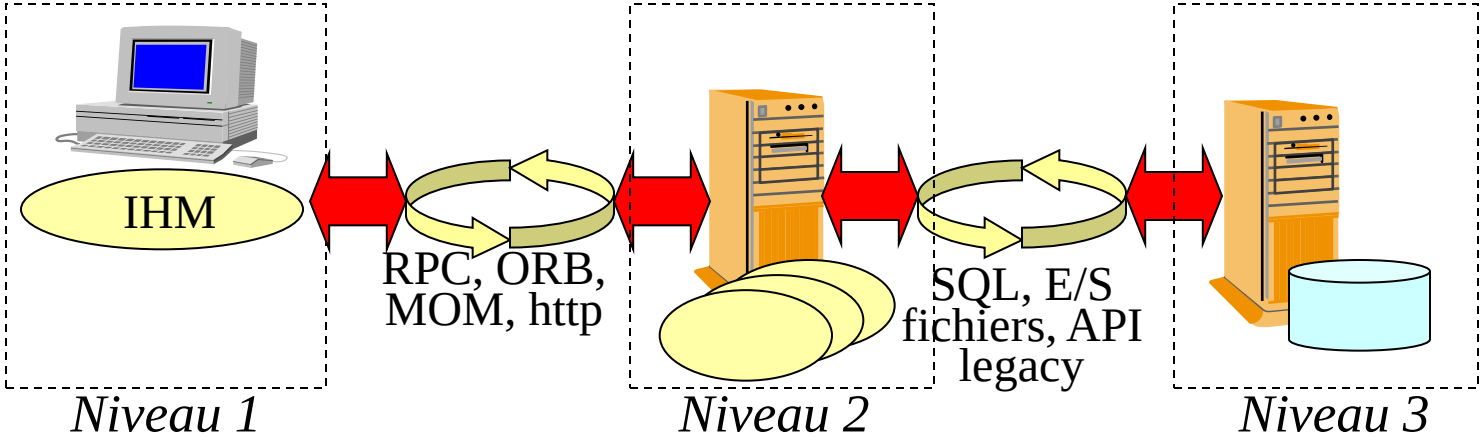
⇨

- réduction du trafic réseau entre postes clients et serveurs
- les applications peuvent être déployées et administrées de manière indépendante des IHM
- placement des serveurs logiciels sur un ou plusieurs serveurs physiques
- nombreuses variantes architecturales possibles

De deux vers trois niveaux (1)

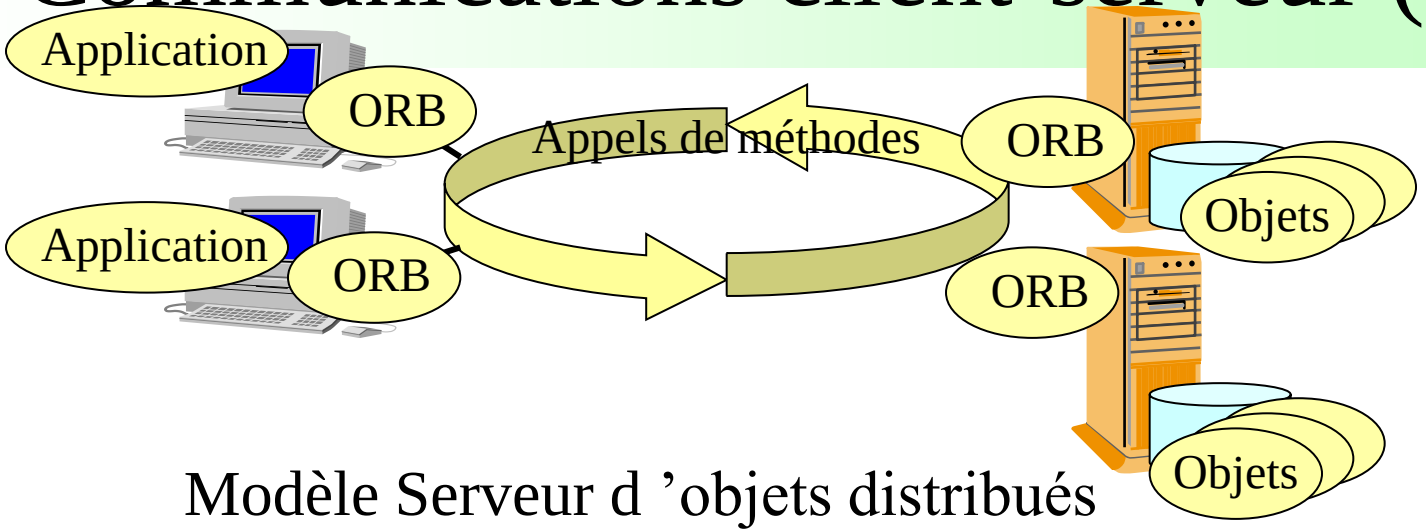


Client serveur à deux niveaux



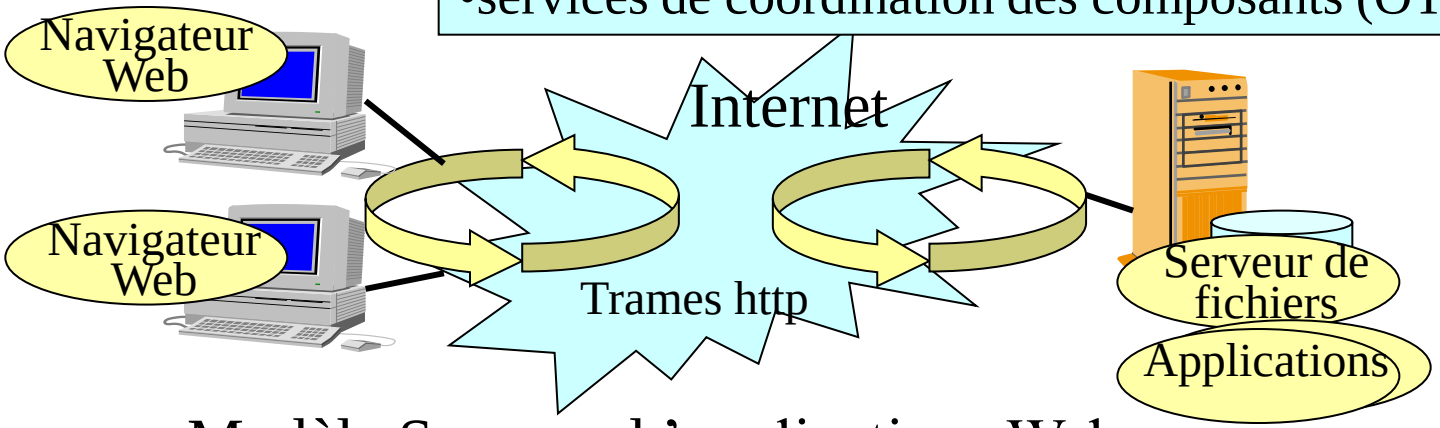
Client serveur à trois niveaux

Communications client-serveur (3)



Modèle Serveur d ’objets distribués

⇨
- Application écrite sous la forme d ’un ensemble d ’objets communicants en interaction
- services de coordination des composants (OTM)



Modèle Serveur d ’applications Web

⇨
- Evolution vers des formes d ’interaction plus élaborées (Web Objet)
- convergence future avec les serveurs d ’objets

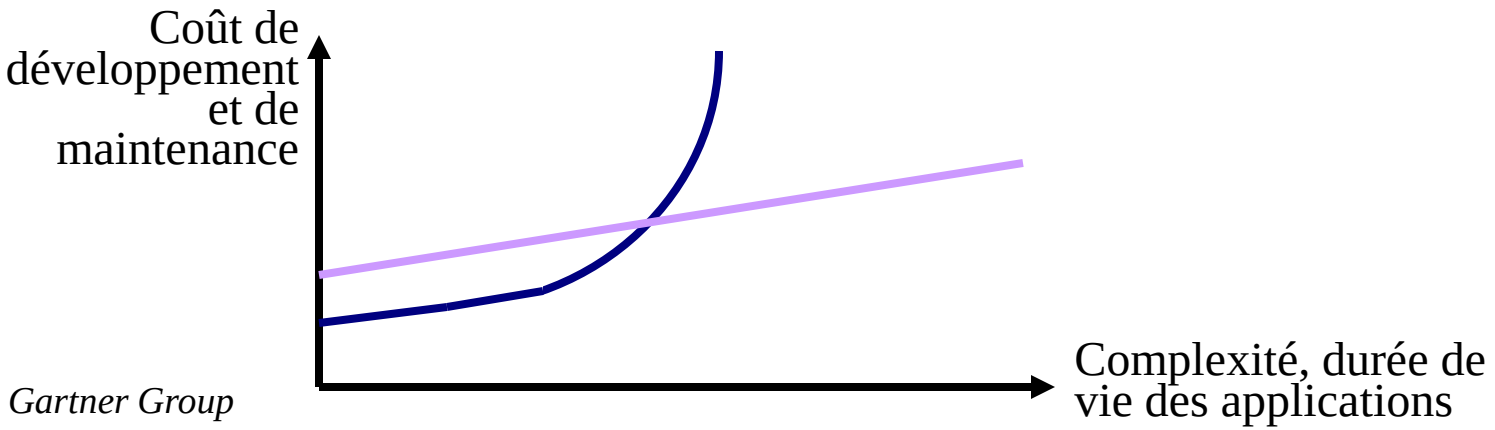
De deux vers trois niveaux (2)

Architecture à deux niveaux

- simplicité, création d ’applications plus rapide
- convient aux applications « départementales »
- difficulté d ’administration et de déploiement
- en général pas extensible aux applications à l ’échelle de la grande entreprise (x 1000 utilisateurs)

Architecture à trois niveaux

- applications mieux dimensionnables, « scalabilité »
- plus faciles à déployer
- adapté aux données issues de sources multiples
- services « abstraits » qui minimisent les échanges sur le réseau
- sécurité (pas d ’exposition du schéma de la BD)



Source : Gartner Group

Modèle à 3 niveaux : applications

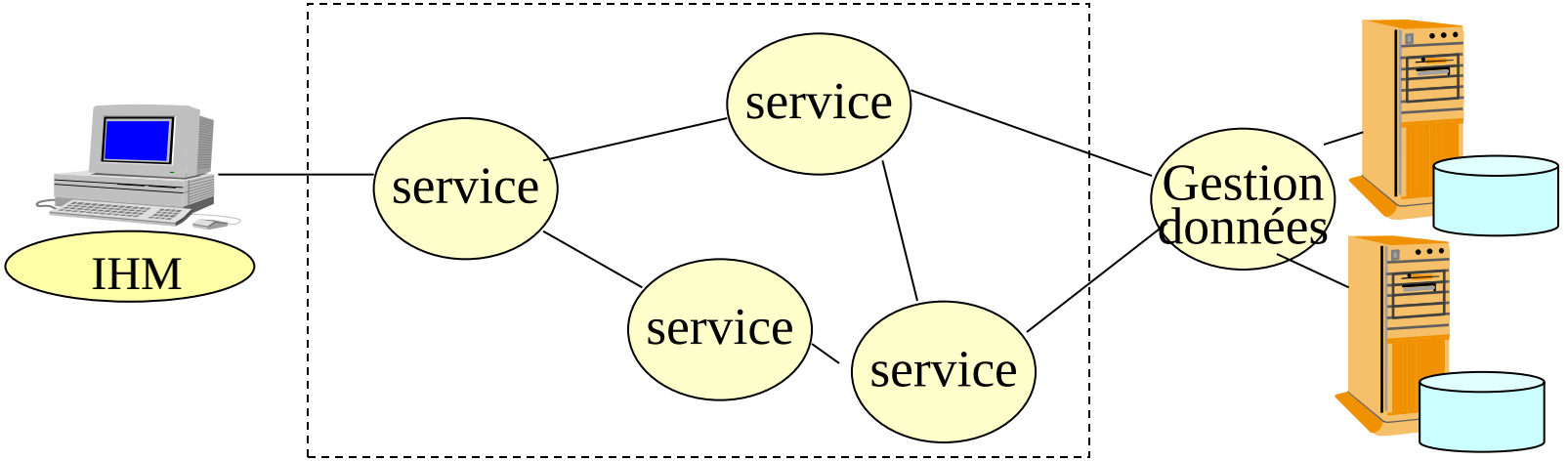
A utiliser si l ’application possède une des caractéristiques suivantes (*selon Gartner Group*) :

- nombreux services (plus de 50)
- applications programmées dans plusieurs langages ou issues de différentes organisations
- sources de données multiples et hétérogènes
- longévité de l ’application > 3 ans
- charge de traitement importante (+ de 300 utilisateurs, + de 50 000 transactions par jour)
- communications inter-applications importantes
- évolution prévue vers l ’une de ces caractéristiques

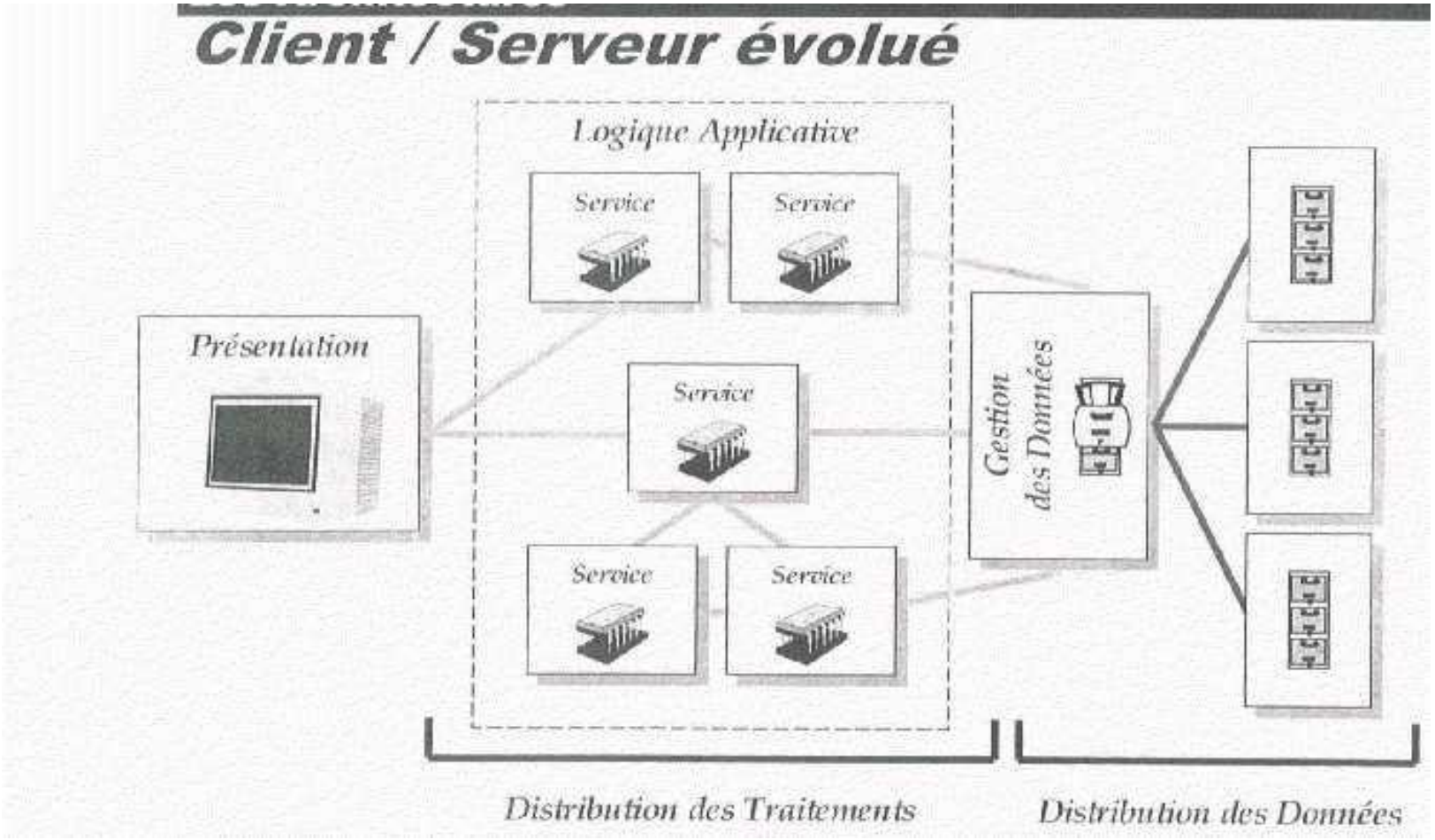
Architecture à n niveaux

- Niveaux intermédiaire ⇨ collection d ’applications
  - chaque application peut être un composant indépendant ayant en charge une fonction
  - chaque fonction peut être utilisée et appeler d ’autres fonctions
  - encapsulation des applications du patrimoine d ’entreprise

⇨ **Architecture à n niveaux**



Architecture par composants

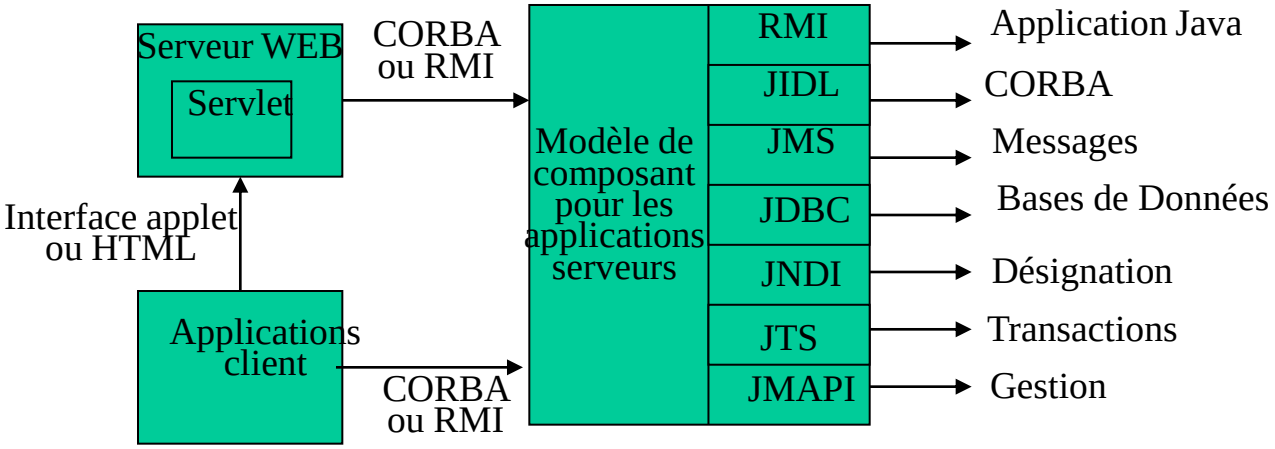


Modèle orienté composant

Enterprise Java Beans (EJB) : modèle de composants pour le développement et le déploiement d ’applications

- partie « serveur » des applications
- parties génériques d ’applications qui peuvent être assemblées pour le construction d ’un système d ’information

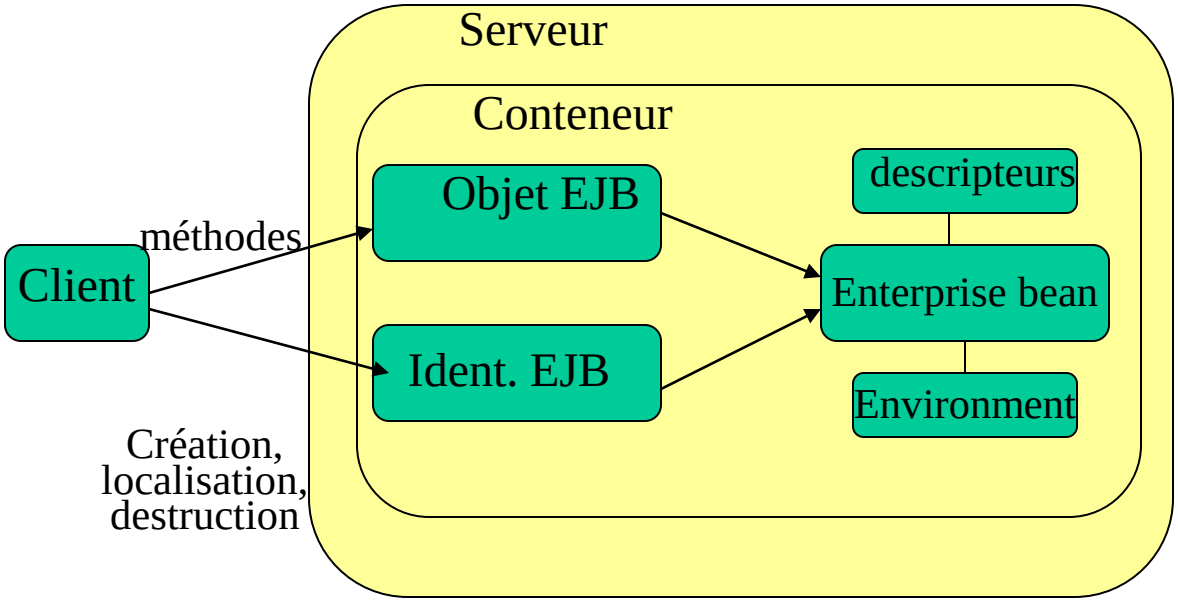
Services de support des EJB



Source : « Serveurs multiprocesseurs, clusters et architectures parallèles ». René Chevance

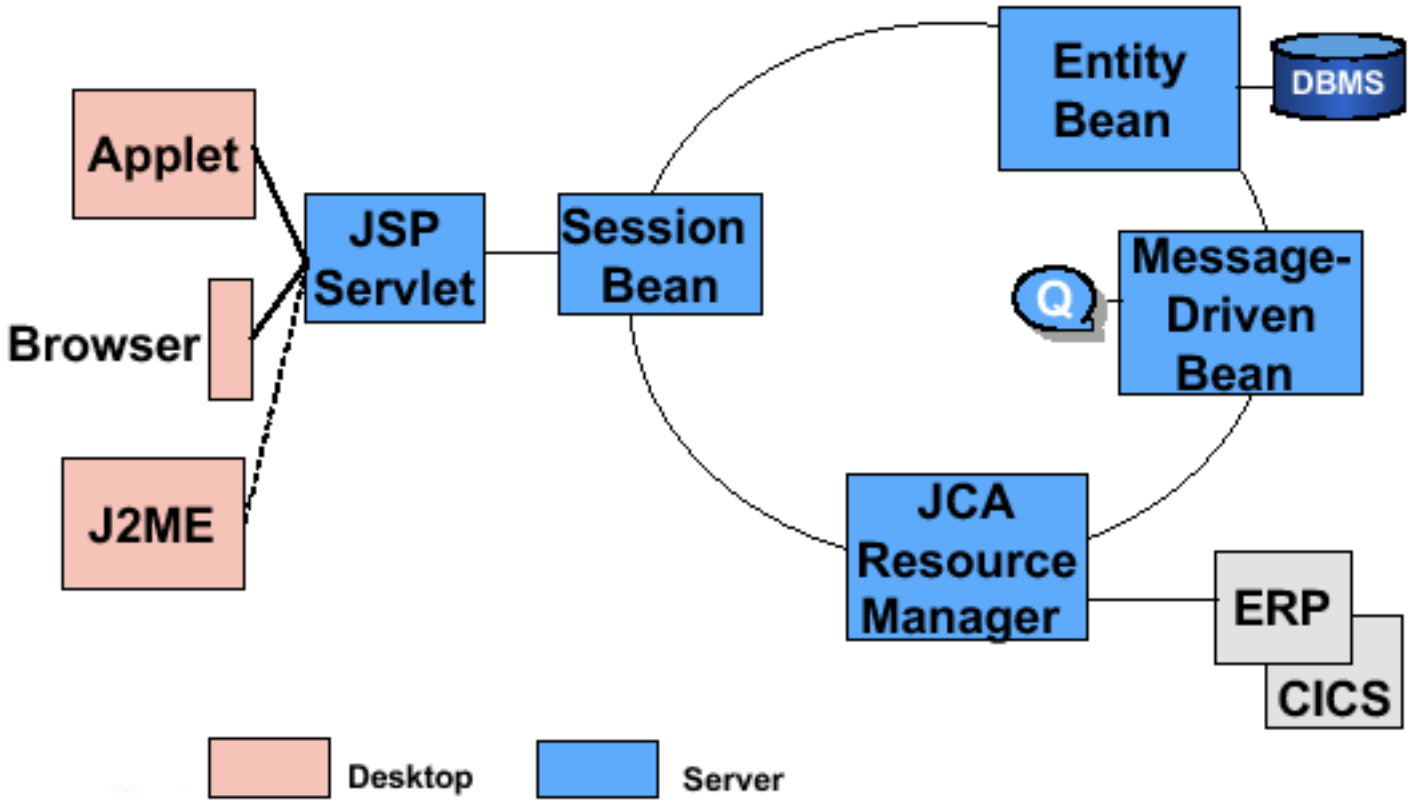
Modèle orienté composant (2)

Structure d ’un conteneur EJB

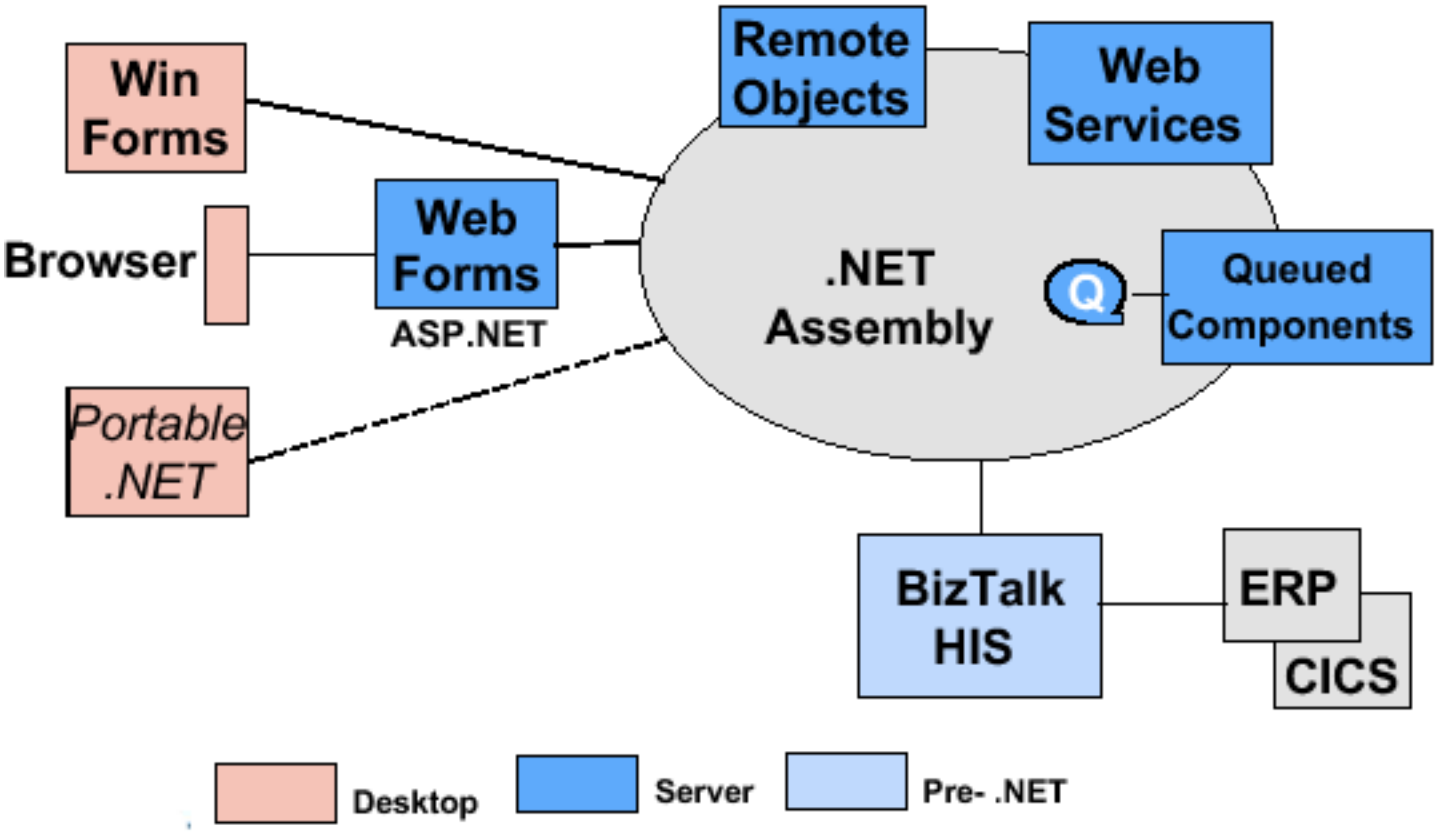


Source : « Serveurs multiprocesseurs, clusters et architectures parallèles ». René Chevance

L'intégration vue par : J2EE



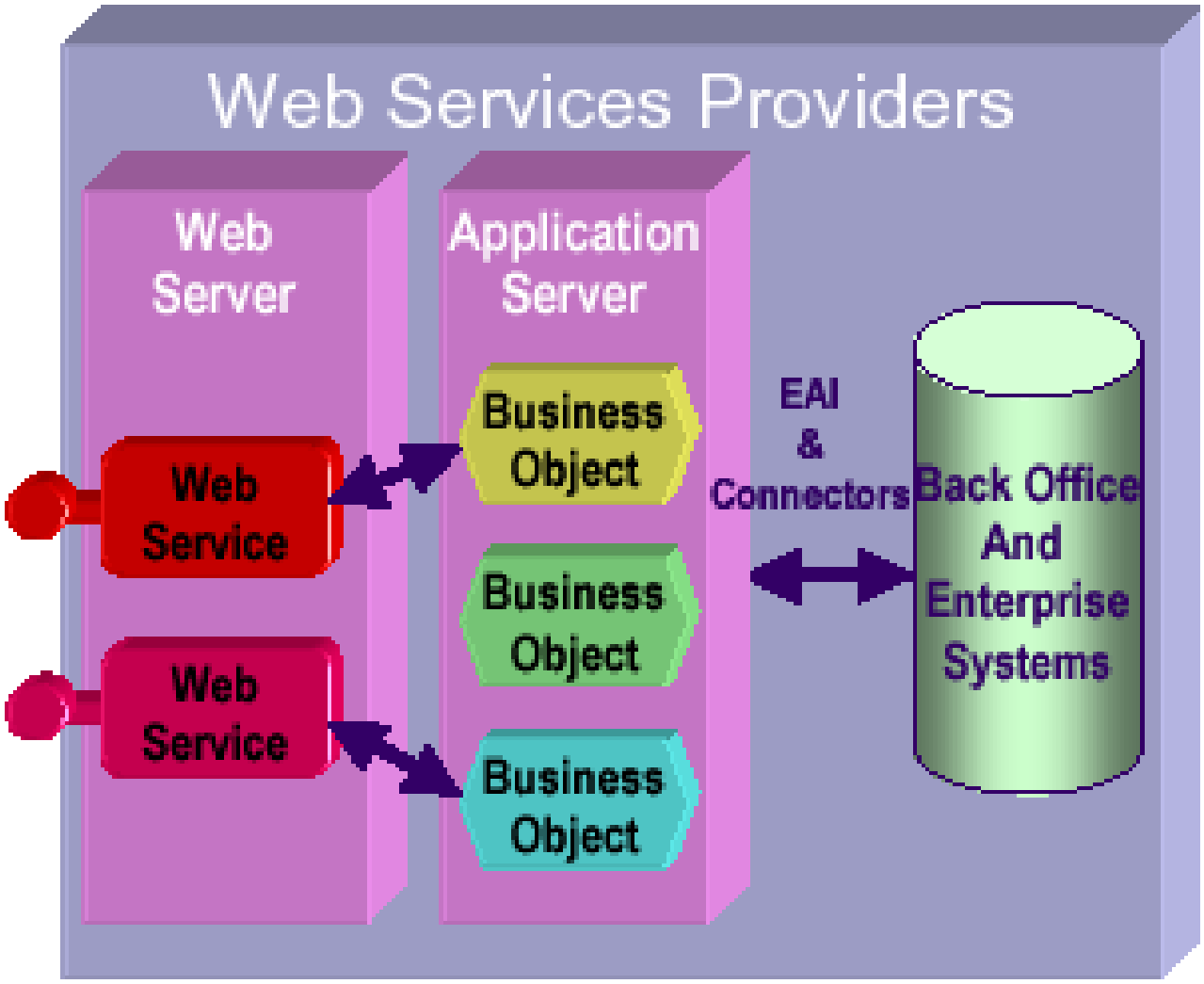
L'intégration vue par : .NET



Les Web Services et SOA

Tout devient « service »

L'accès universel : Web services



Didier Donsez, Web Services

XML comme structure d'échanges

- Standard W3C et OASIS
- La syntaxe XML est très simple.
- XML est un méta-langage donc extensible, on peut rajouter des balises.
- XML est indépendant des plates-formes: Portabilité
- Outils disponibles
- **Largement utilisé pour les échanges inter-applications**

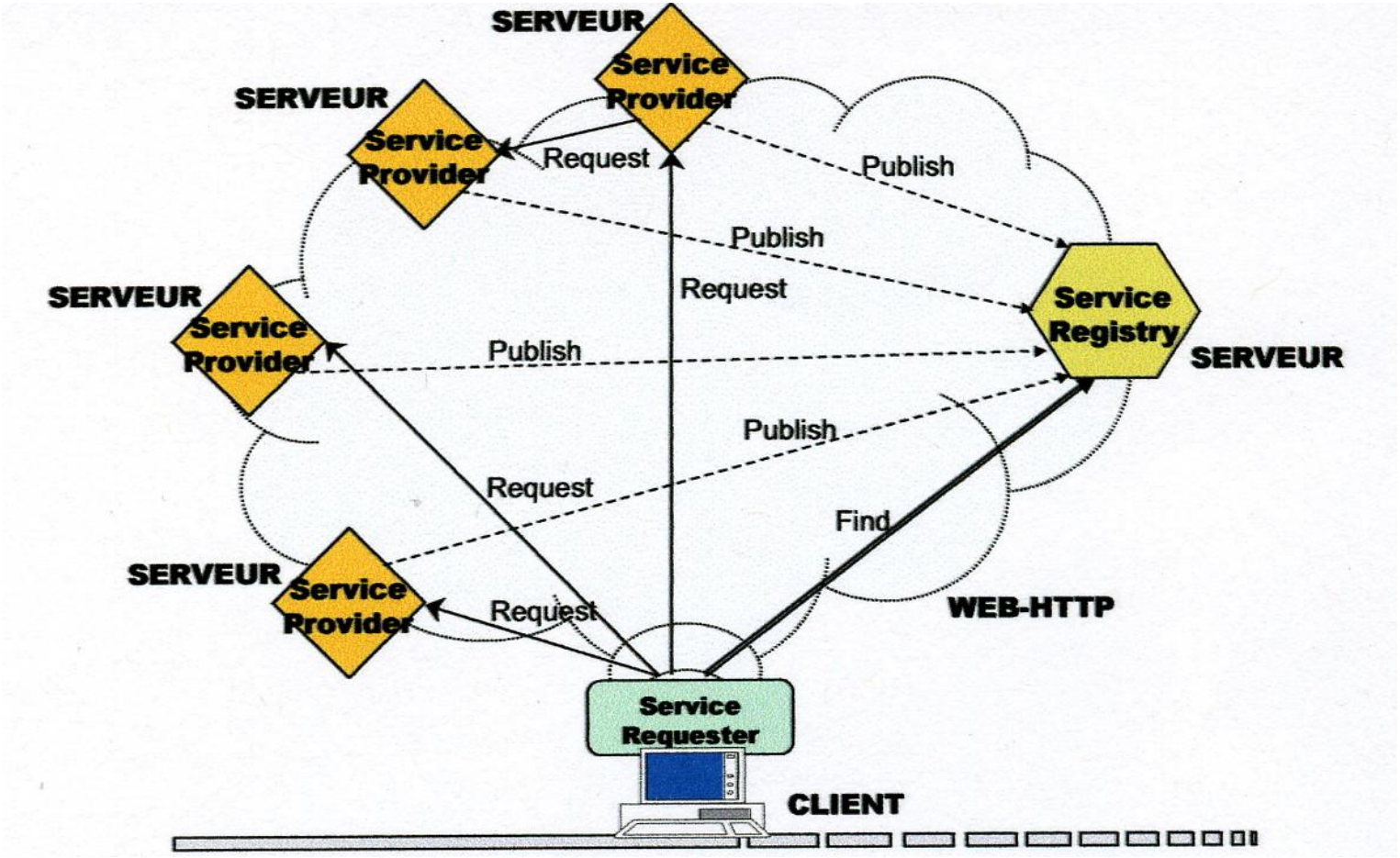
Les Services Web

- Objectifs : accès rapide à l'information, Système ouvert réduisant les coûts, administration simplifiée ⇨ utilisation d'Internet comme support de communication
- Web Service : unité logique applicative accessible en utilisant les protocoles standards d'Internet, réutilisable, et indépendante de la plate-forme et de l'implémentation

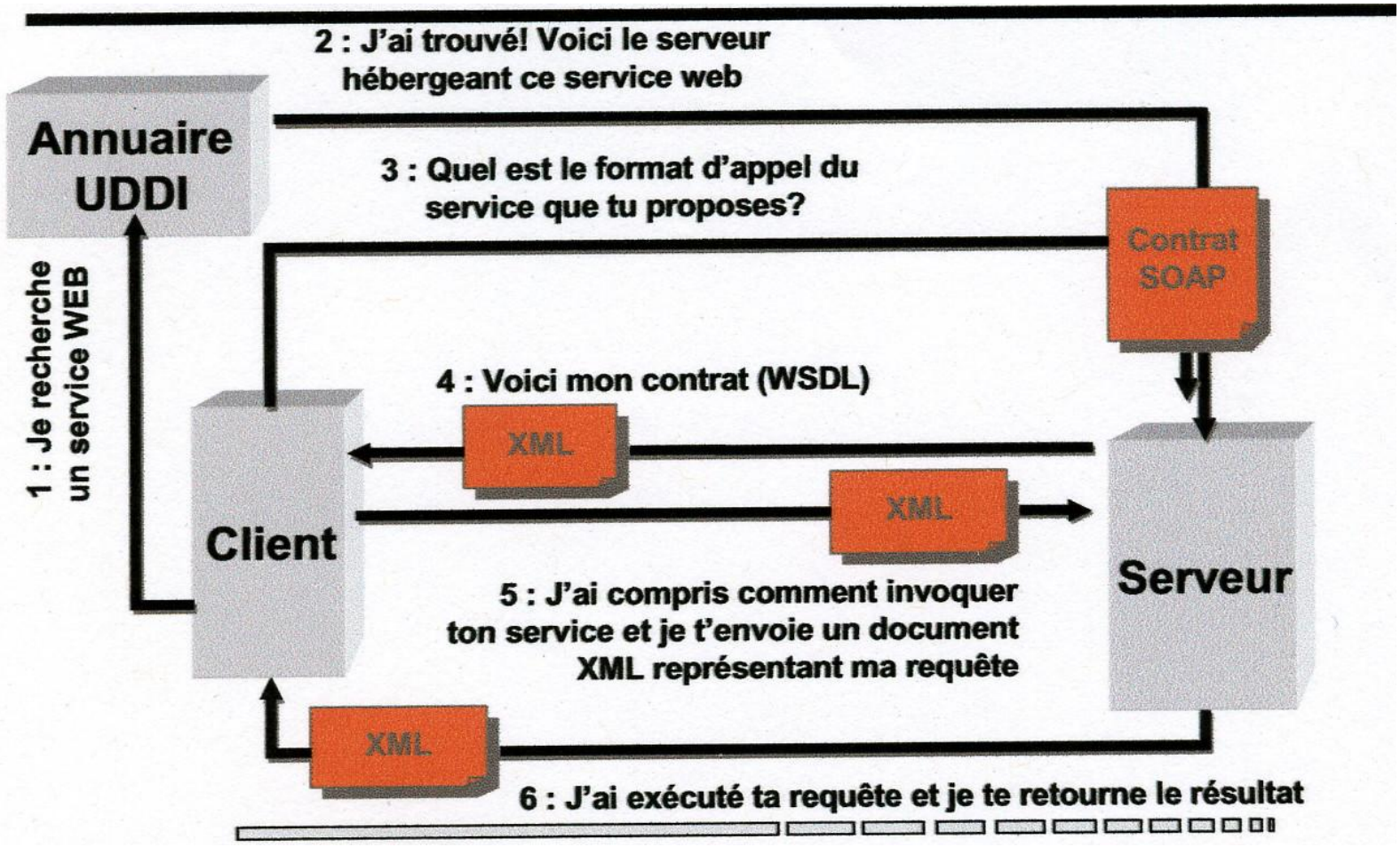
Caractéristiques des Services Web

- Application fournissant des traitements et des données à d'autres applications déployées sur le Web, et vues à travers une interface
- protocole SOAP au dessus de http
- Données en XML
- Répertoire de services : UDDI
- Langage WSDL pour décrire le service

Les Web Services. Principes



Les Web Services. Principes (2)



Conclusion

- Des technologies multiples,
- Différents modèles d'architectures
- Différents « middlewares » ou « intergiciels »
- Une évolution vers l'intégration de l'hétérogène à travers la distribution à travers Internet
- Le nouveau modèle : l'orchestration de services