Cours 420-B56-GG

Bases de données III

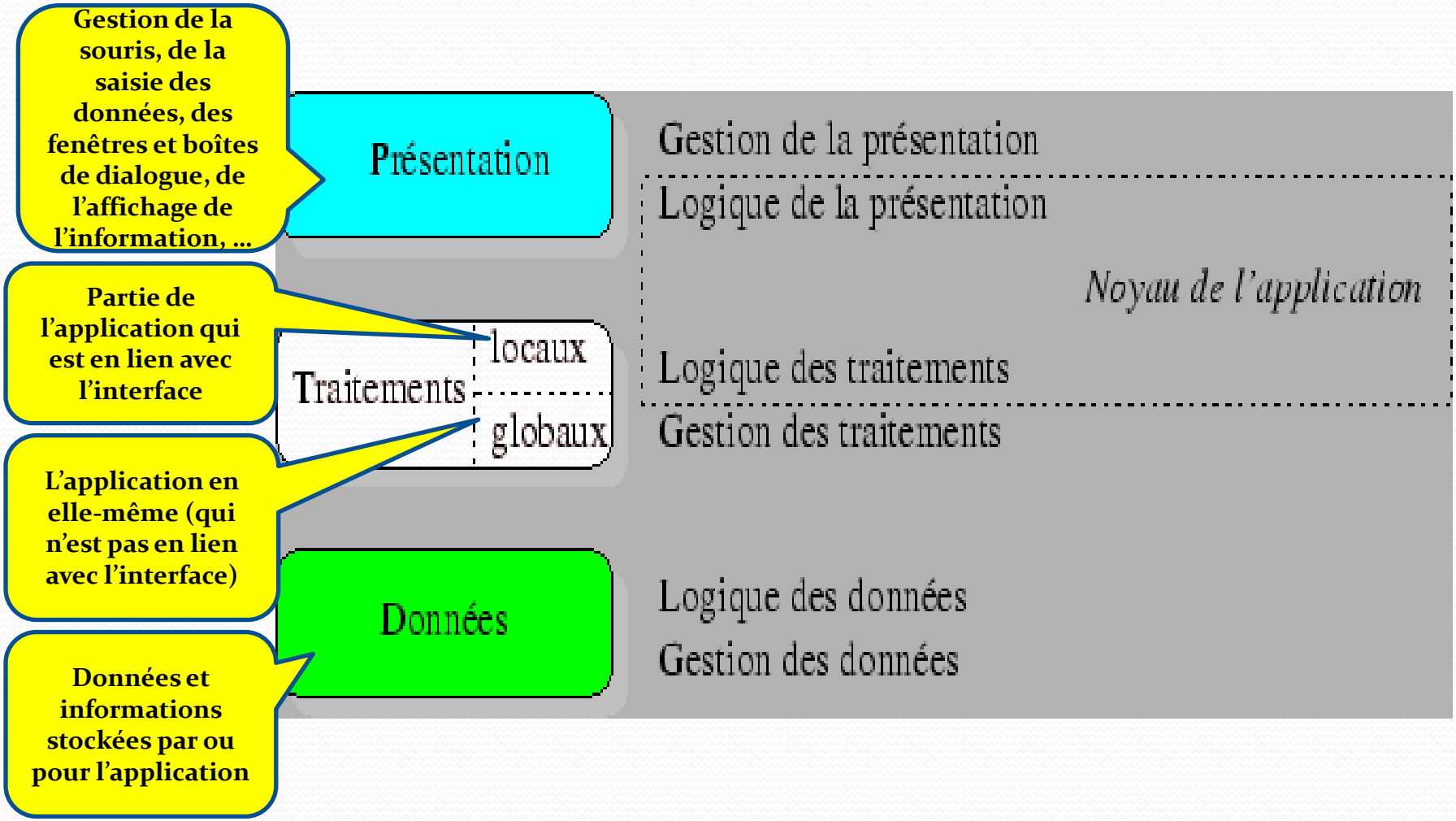
Les architectures d'application

Les trois couches d'une application (1)

Dans chaque application, on retrouve 3 couches:

- La couche **Présentation** (ou l'interface) :
  - Cette couche représente le côté visuel et interactif de l'application. C'est le lien entre l'utilisateur et l'application.
- La couche **Traitements** (ou logique applicative) :
  - Cette couche représente le cœur de l'application. Ce sont les différentes fonctionnalités de l'application.
- La couche **Données**:
  - Cette couche assure la gestion des données et des informations stockées par ou pour l'application.

Les trois couches d'une application (2)



L'architecture d'une d'application

- Les trois couches d'une application peuvent être réparties de différentes manières entre plusieurs ordinateurs (machines physiques).

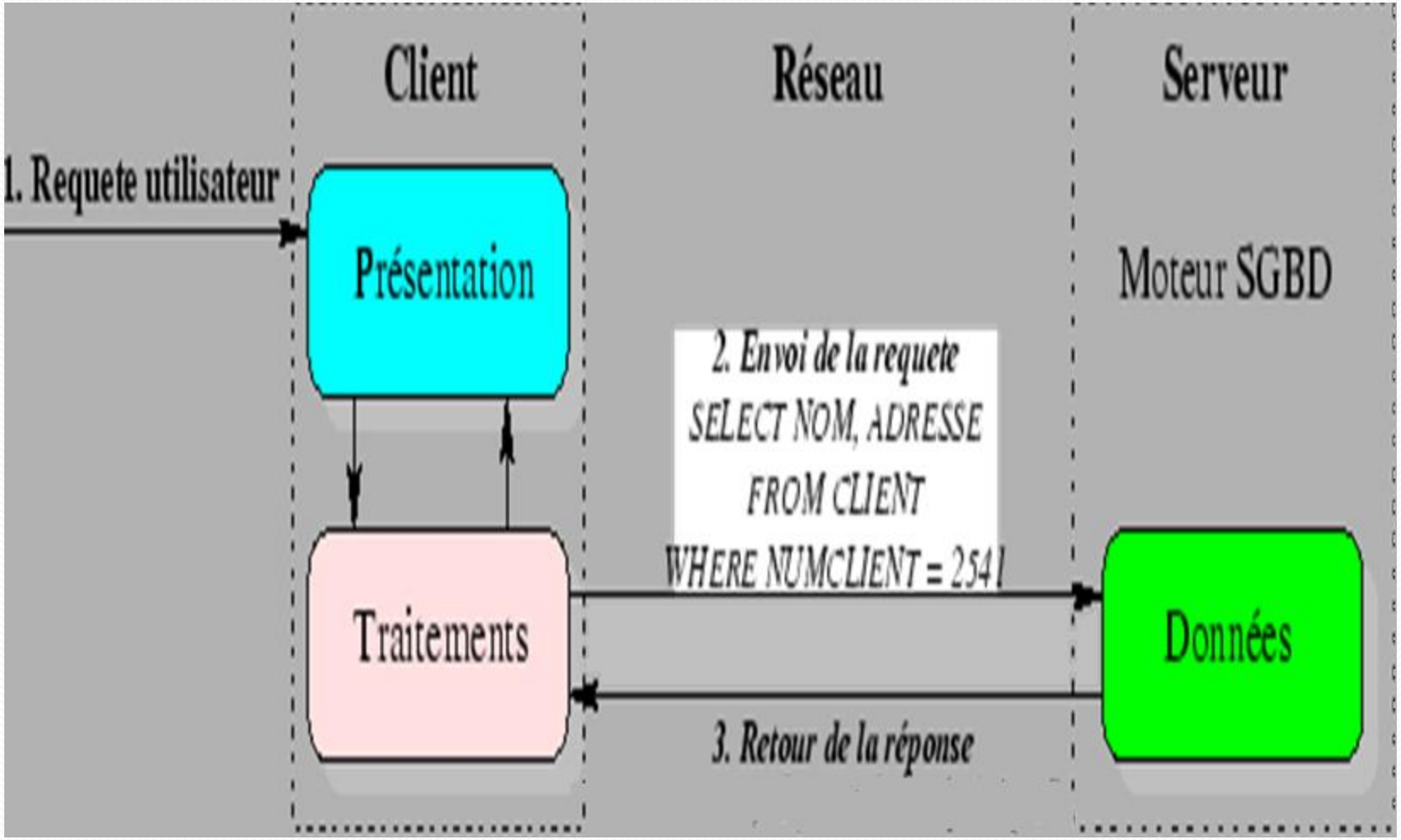
- La répartition des différentes couches permettent de distinguer différents types d'architecture : 1-tier, 2-tiers, 3-tiers et n-tiers

L’architecture 1-tier (centralisée)

- Les trois couches sont intimement liées et s'exécutent sur le même ordinateur. On parle ici d’architecture centralisée ou autonome.
- Dans le contexte d’un utilisateur isolé, cela ne cause aucun problème, mais peut s’avérer problématique dans un contexte multi-utilisateurs.
- Dans un contexte multi-utilisateurs, plusieurs utilisateurs partagent des données stockées sur un ordinateur commun.
- Dans ce contexte, cela peut engendrer des problèmes d'administration et de sécurité des données car la gestion des données fait partie intégrante de l’application.

Architecture 2-tiers (client-serveur)

- La gestion des données est centralisée et est assurée par **un moteur de base de données** installé sur un serveur nommé *serveur de données* (par exemple, ***SQL Server***, ***mySql***, etc…)
- La couche présentation (interface utilisateur) et la couche traitements sont situés sur le poste client.
- La complexité et la puissance du traitement concernant les données dans la base de données sont à la charge du serveur. Le poste client est plus léger.


Client
Réseau
Serveur
1. Requete utilisateur
Présentation
Moteur SGBD
2. Envoi de la requete
SELECT NOM, ADRESSE
FROM CLIENT
WHERE NUMCLIENT = 2541
Traitements
Données
3. Retour de la réponse

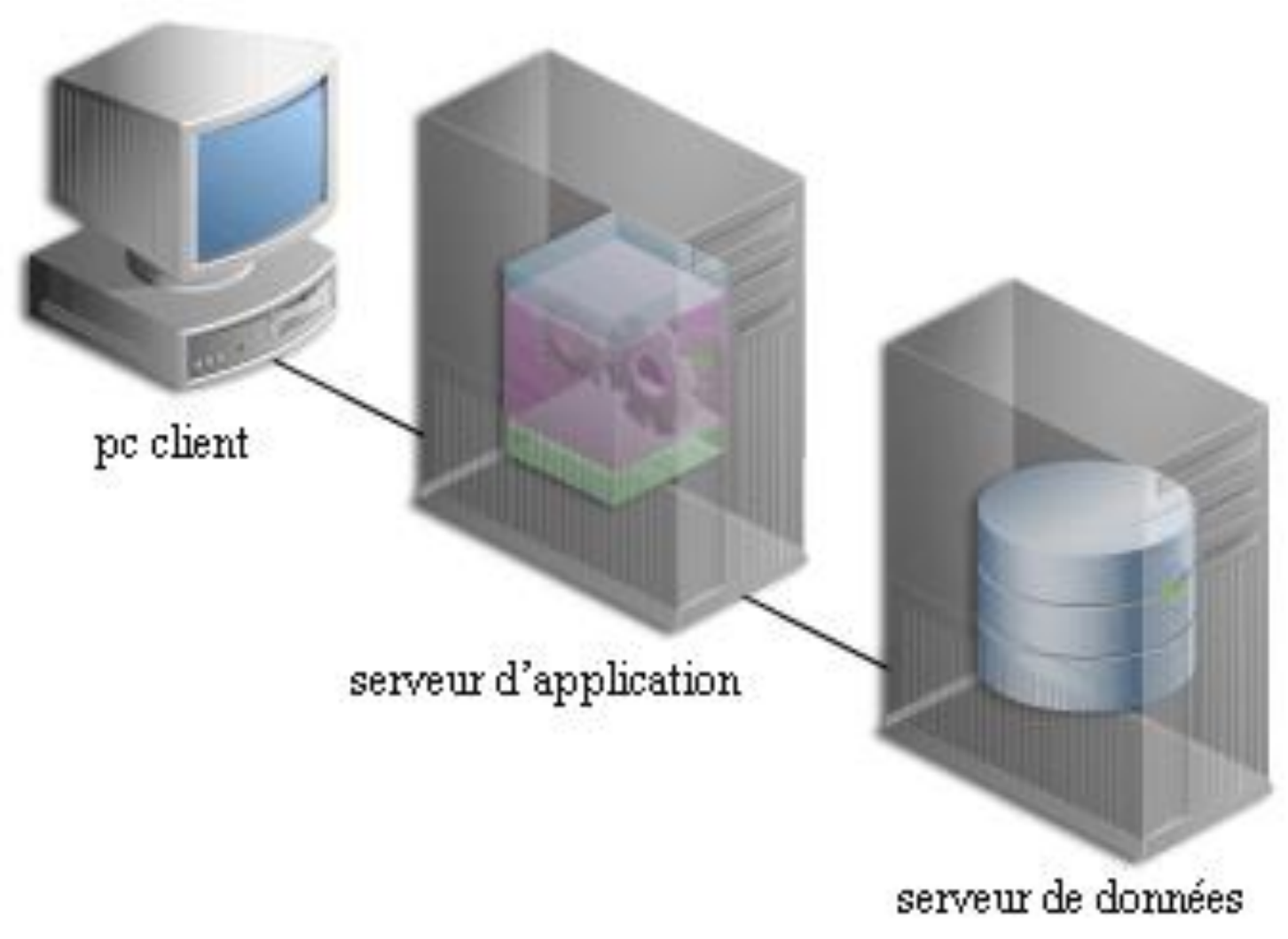
Limites de l’architecture client-serveur

- Si trop de clients veulent communiquer avec le serveur au même moment, ce dernier risque de ne pas supporter la charge.
- Si le serveur n'est plus disponible, plus aucun des clients ne fonctionne.
- Les coûts de mise en place et de maintenance peuvent être élevés.
- La relation étroite qui existe entre le client et l'organisation de la partie serveur complique les évolutions de cette dernière (il faut que les versions correspondent).
- Les mises-à-jour sont fréquentes.

Architecture 3-tiers (Distribuée)

- La présentation (l'interface) est prise en charge par le poste client.
- La logique applicative est prise en charge par un serveur intermédiaire nommé *serveur d'application*.
- Les données sont gérées de façon centralisée par le *serveur de données*.
- On parle ici d'**architecture distribuée**.

Architecture 3-tiers



Avantages de l’architecture distribuée

- Elle sépare nettement toutes les couches de l'application.
- Le poste client est plus "léger" en ce sens qu'il ne contient aucun traitement applicatif. Il ne contient que l’interface.
- Donne plus de flexibilité et de souplesse à l'application.
- S'adapte mieux à la mise en place des nouvelles technologies.
- D'un point de vue développement, la séparation qui existe entre le client, le serveur d’application et le serveur de données permet une spécialisation des développeurs sur chaque tiers de l'architecture.
- Notez qu’il peut y avoir plusieurs serveurs d’application différents (et parfois plusieurs serveurs de données différents); ce type d’architecture se nomme **architecture n-tiers**.

Limites de l’architecture distribuée

- Les serveurs constituent la pierre angulaire de l'architecture; le client a une charge de travail beaucoup moins lourde, mais les serveurs sont fortement sollicités. Il est très difficile de bien répartir la charge de travail entre clients et serveurs.
- De plus, les solutions mises en œuvre sont relativement complexes à maintenir et la gestion est compliquée. Cela demande une bonne expertise au niveau du développement.
- Finalement, les coûts au niveau du développement et de la maintenance sont très élevés.