Architecture des systèmes d'information

Table des matières

1 La décennie 70 1

2 Le modèle relationnel (les années 80) 2

3 Enrichissement du relationnel (les années 80/90) 2

4 Système d'informations (les années 80/90) 3

5 Le micro-ordinateur (fin des années 80) 3

6 Le client/serveur (les années 90) 3

7 Architecture à deux niveaux (2-tier) (années 2000) 4

8 Architecture des applications 4

9 Architecture à trois niveaux (3-tier) 5

10 Le client reprend de l'importance 5

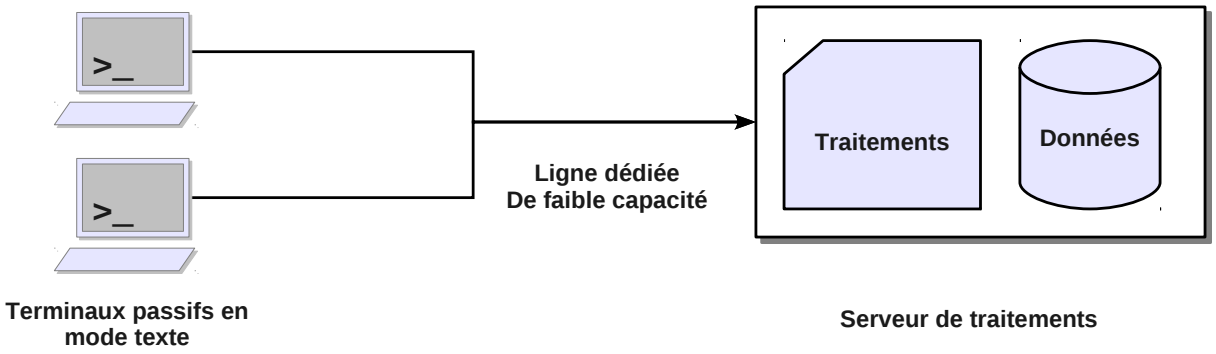
11 Architecture orientée services (SOA) 6

12 Architecture orientée mico-services 6

13 Architecture JEE 7

14 JEE « outillé » 7

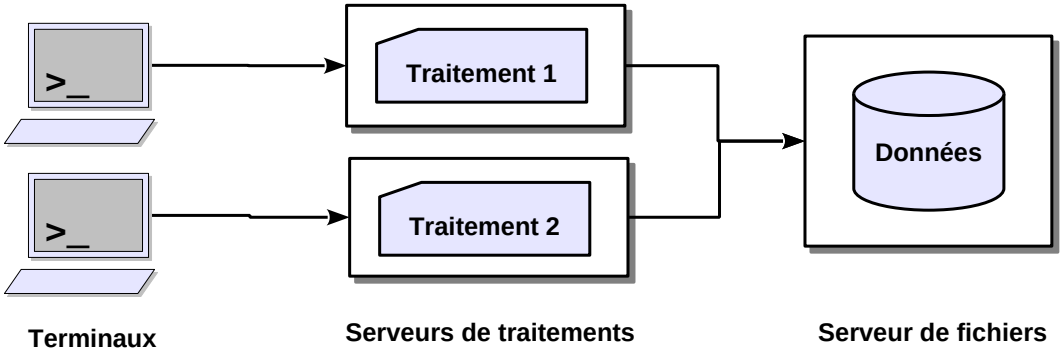
1 La décennie 70



Le **service informatique** gère des serveurs de traitements qui alimentent des terminaux.

Les traitements et les données sont **centralisés** sur des serveurs d'infrastructure.

Les **données** sont éventuellement centralisées sur des serveurs spécifiques (serveurs de fichiers).



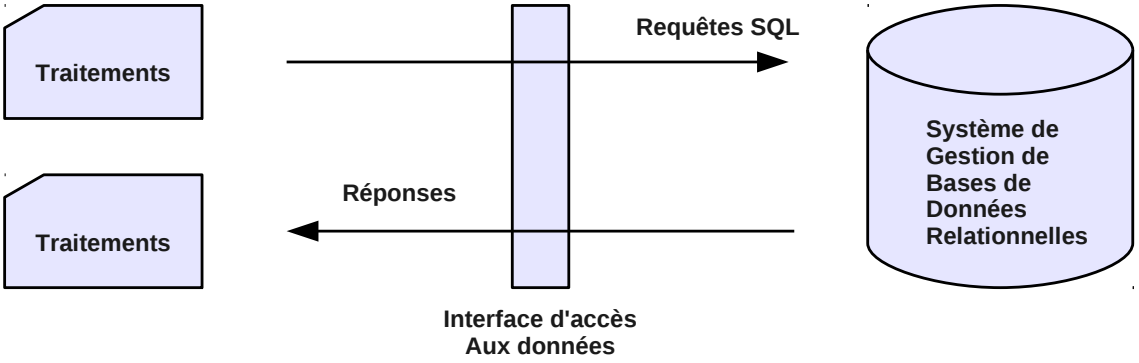
Cette architecture permet

- de sécuriser les données
- d'éviter les redondances

2 Le modèle relationnel (les années 80)

1970 : CODD « *Relational model of data* »

1982 : Création d'Oracle



3 Enrichissement du relationnel (les années 80/90)

- L'interface d'accès aux données s'améliore :
  - ▷ Création des vues (simplification du modèle),
  - ▷ Création des triggers (introduction des traitements),
  - ▷ Création des procédures stockées (amélioration des traitements),
  - ▷ Définition des utilisateurs et des rôles (sécurisation),
- La gestion des données est la partie **la plus importante**.
- Mise en place des méthodes de conception **orientées données** :
  - ▷ Entités / Associations
  - ▷ Merise / Modèle Conceptuel des Données
- Création du métier d'**administrateur des données (DBA)**.

4 Système d'informations (les années 80/90)

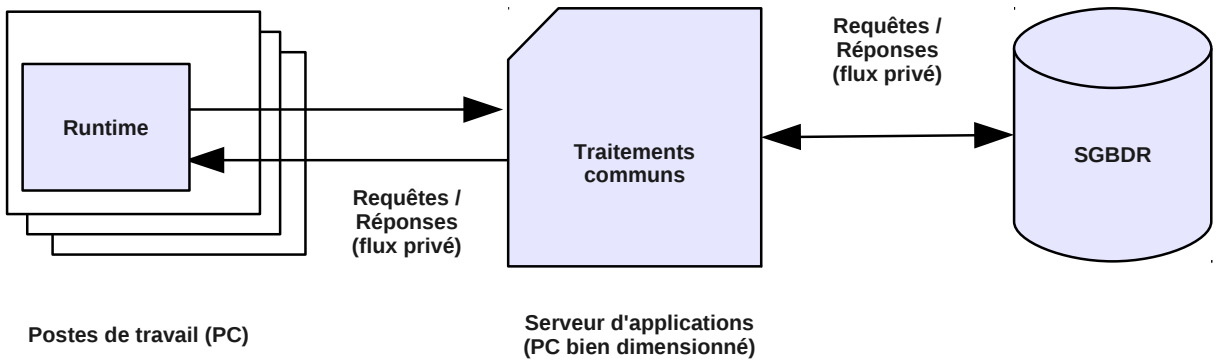
- Le centrage sur les données permet :
  - ▷ une approche globale (toutes les applications **partagent** et **modifient** les mêmes données),
  - ▷ l'activité de l'organisation est modélisée et représentée par les données (approche systémique).
- Le **service informatique** est remplacée par le **système d'informations (SI)**

  SI : un ensemble **structuré de ressources** (matériels, logiciels, données) qui permet la collecte, le traitement et la distribution des informations de l'organisation.
- L'architecture des SI est un problème complexe.

5 Le micro-ordinateur (fin des années 80)

- Caractéristiques :
  - ▷ coût important
  - ▷ faible capacité de calcul et de stockage
- Problèmes posés par le micro-ordinateur :
  - ▷ Coordination avec l'informatique d'entreprise (et le service informatique)
  - ▷ Luttes de pouvoir pour savoir où sont les données et les traitements
- Conséquences :
  - ▷ Apparition d'applications autonomes
  - ▷ Duplication (avec erreurs) des données entre le SI et les PC

6 Le client/serveur (les années 90)



Avantages :

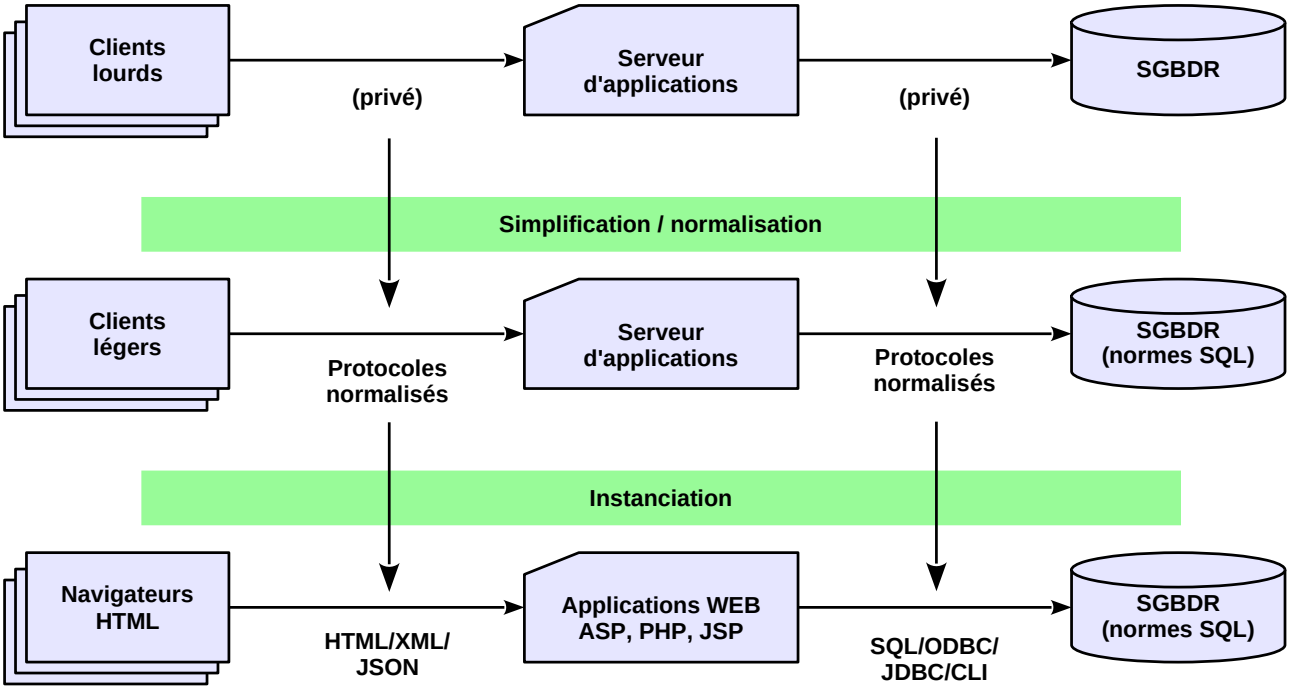
- Serveur de taille moyenne (baisse du cout),
- Intégration des PC (hétérogènes) dans le SI.

Inconvénients :

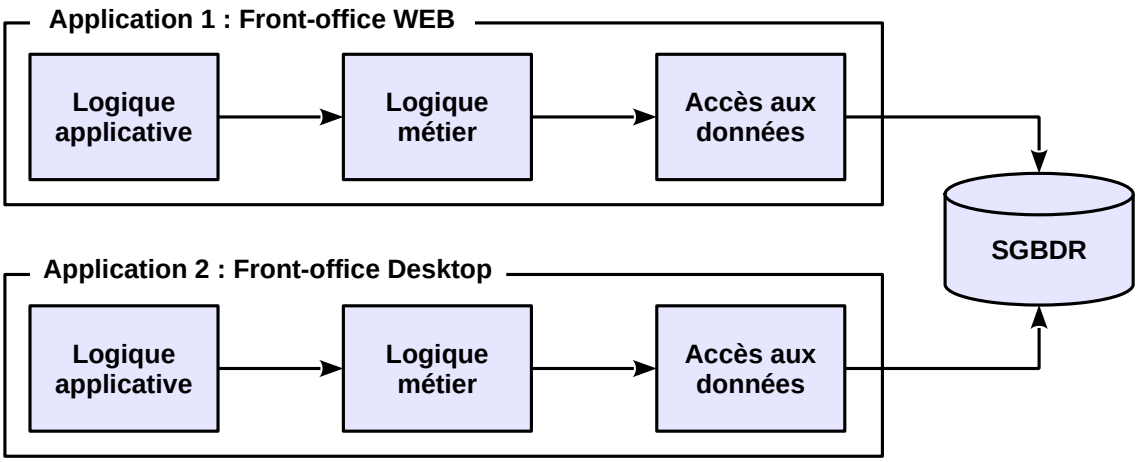
- Le poste de travail est qualifié de **client lourd**,
- Le coût d'exploitation est très important (poste difficile à maintenir),
- Pas de porte ouverte vers les traitements et les données.

7 Architecture à deux niveaux (2-tier) (années 2000)

Objectif : Simplifier et normaliser le client/serveur.



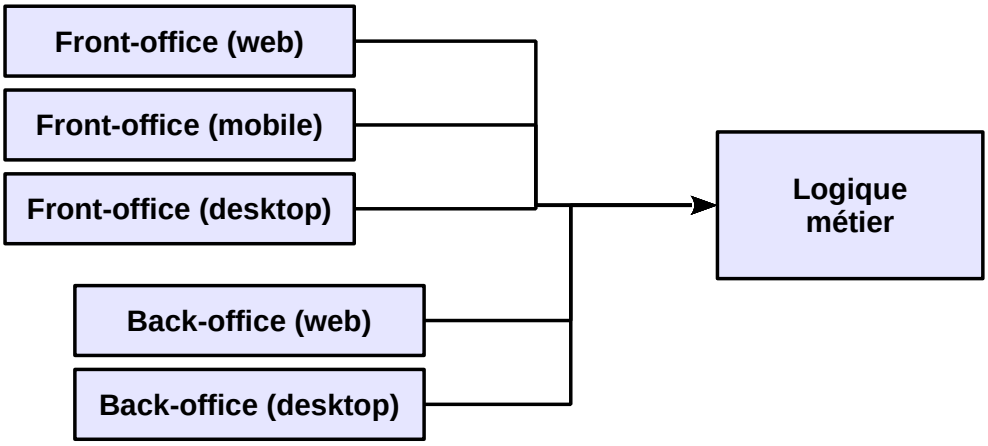
8 Architecture des applications



Constats :

- Les parties **métier** et **données** sont dupliquées dans plusieurs applications,
- Il est très difficile de maintenir ces implantations (technologies, ressources humaines, objectifs différents),
- La couche de stockage est le seul élément commun.

La couche métier regroupe **les actions métier** qui sont indépendantes des logiques applicatives :

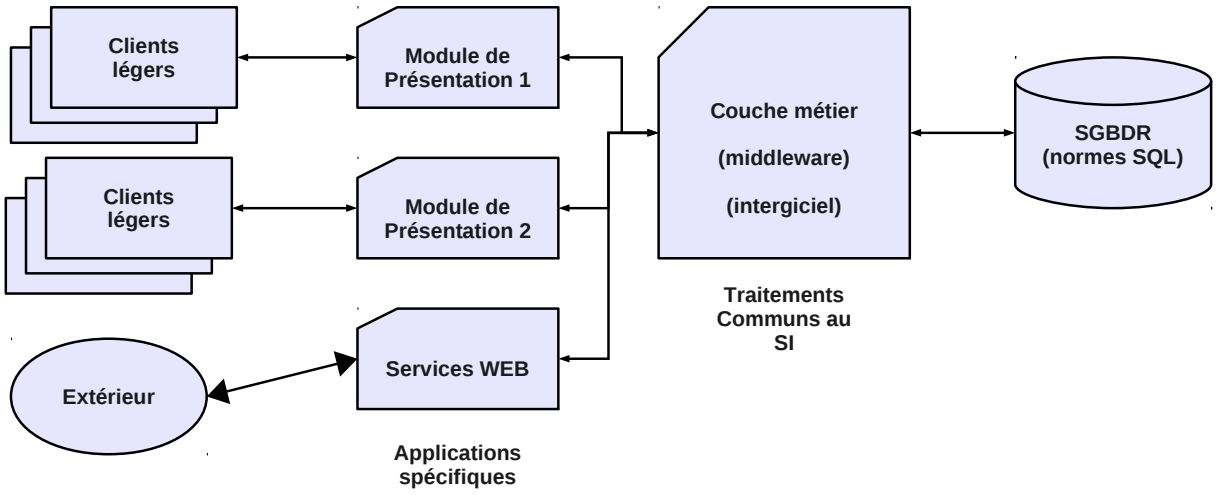


Cette approche permet de

- simplifier les applications,
- gérer les communications inter-applications,
- pérenniser et sécuriser et faire évoluer la partie métier.

9 Architecture à trois niveaux (3-tier)

Introduction du **middleware** qui permet de **factoriser** les traitements métier.



Avantages :

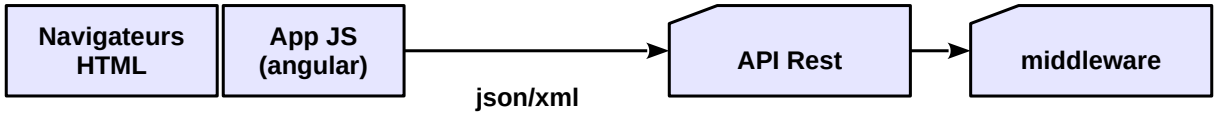
- Simplification des applications
- Indépendance du stockage
- Montée en charge
- Sécurité accrue
- Ouverture vers les traitements
- N-tiers

Inconvénients :

- Choix d'une technologie
- Choix d'une couche de transport
- Conception du middleware
- Empilement des couches

10 Le client reprend de l'importance

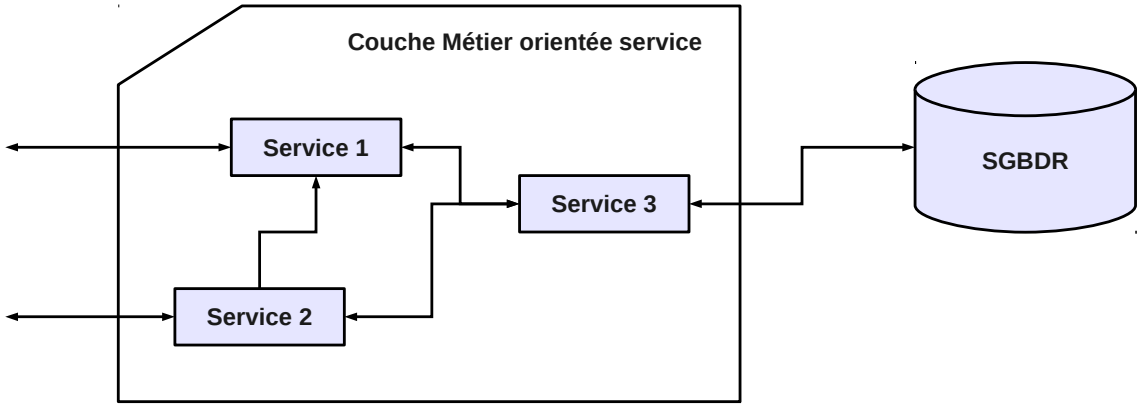
Dans les applications récentes, nous allons **délocaliser** vers le client une partie des traitements (code javascript) :



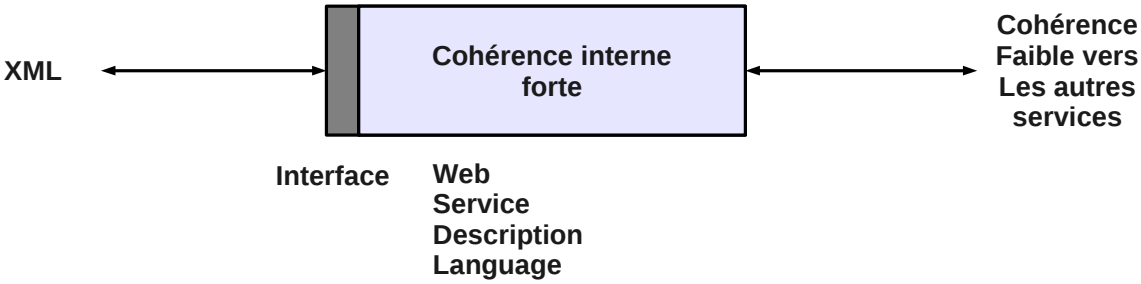
Avantages :

- meilleure interactivité des applications,
- diminution du traffic réseau,
- utilisation des ressources locales,

11 Architecture orientée services (SOA)



Définition d'un service logiciel (brique de base) :



On utilise des principes d'orchestration (définition de processus métiers basés sur des services métiers) (BPEL : Business Process Execution Language).

12 Architecture orientée mico-services

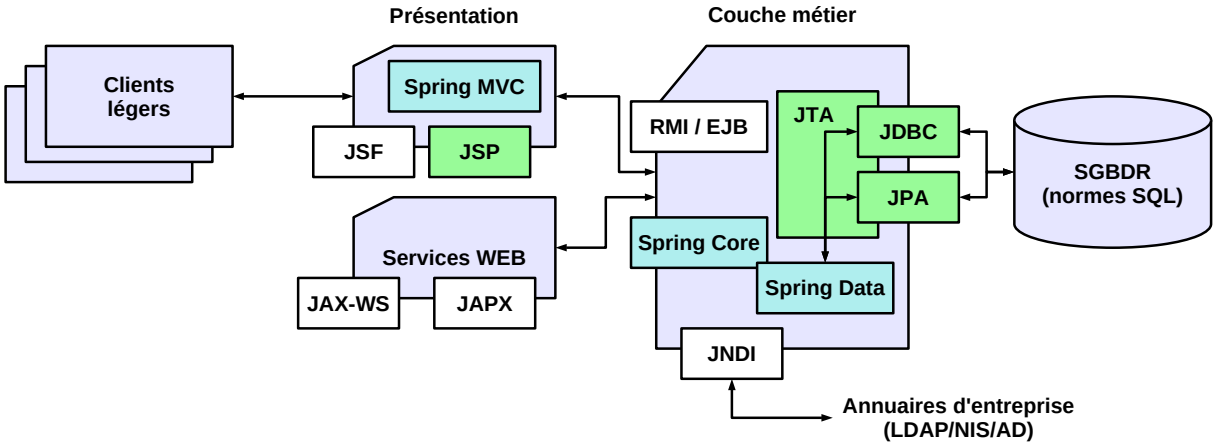
Avant : Un **service** = un métier (achat, vente, réservation). Un service est donc un objet complexe difficile à faire évoluer.

Maintenant : Mise à place de **micro-service** :

- Un service = un sous-métier, une entité
- Avantages :
  - ▷ simplification des services,
  - ▷ augmentation du découplage,
  - ▷ conception, réalisation, déploiement facilités (agilité),
  - ▷ utilisation des conteneurs,
  - ▷ l'orchestration est de plus en plus importante.

13 Architecture JEE

Java Enterprise Edition : J2EE 1.1 puis 1.4 puis JEE 5 puis JEE 6, JEE 7 (2013), Jakarta EE 8 (2017) et JEE 9 (2020).

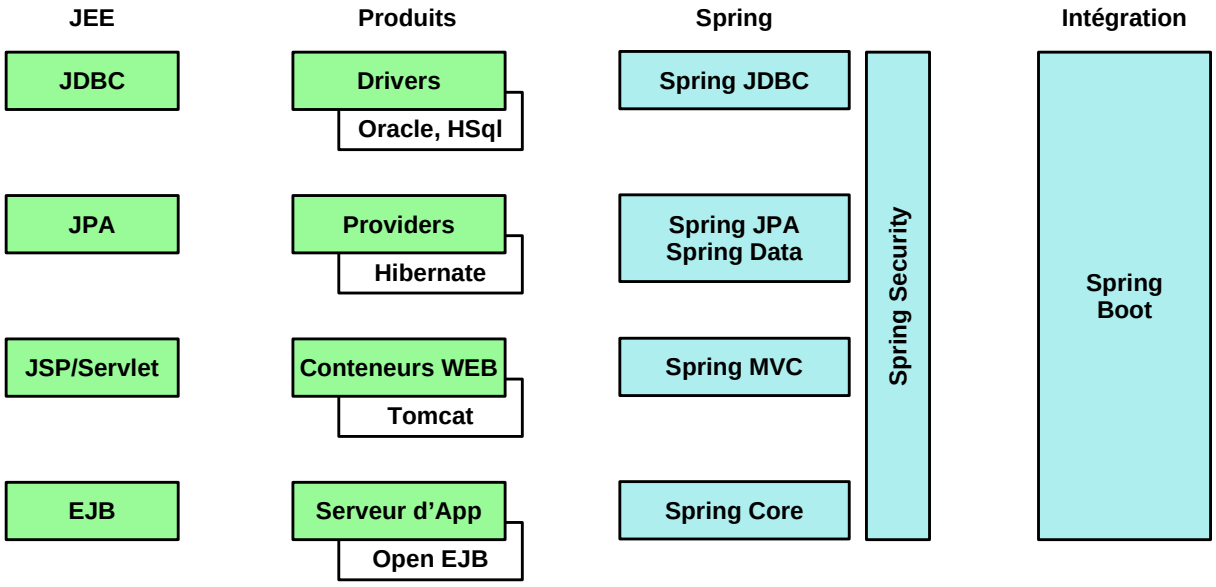


JEE : ensemble de **spécifications** destinées à définir un environnement multi-niveaux (n-tiers) pour des applications réparties d'entreprises.

Cours : Architecture, Spring, JDBC, JPA, Servlet/JSP, Spring MVC

14 JEE « outillé »

Dans les **applications modernes**, la plateforme JEE a besoin d'être outillée.



Sprint Boot est un facilitateur qui permet de mettre en cohérence des composants d'origine variée.