MODULE 6

Applications n-tiers

HUMAN booster
VOTRE SOLUTION COMPETENCE

Les applications n-tiers

Plan du module “Applications n-tiers”

- Les architectures n-tiers
- La couche Persistance
- La couche Métier
- La couche Présentation
- Pour approfondir

Les applications n-tiers

Les architectures n-tiers

Un peu de terminologie

Qu’est-ce que l’architecture d’un système **informatique** ?

- Ensemble des composants matériels et logiciels qui forment le système
  - Serveurs, postes client, routeurs et switchs, pare-feu, NAS, KVM, ...
  - Systèmes d’exploitation, classes développées, librairies et exécutables externes (COTS : SGBD, serveur Web, librairie cryptographique, ...)
- Les relations entre ces composants (installation, inclusion, activation, communication, ...)

Un peu de terminologie

Qu’est-ce que l’architecture d’un système **logiciel** ?

- La liste des éléments logiciels qui le composent
  - Systèmes d’exploitation, code développé, librairies et exécutables “tierce partie” (COTS), etc.
- Les relations entre ces éléments (association, inclusion, dépendance, communication, ...)
- Les activités (processus et threads) dans lesquelles interviennent ces éléments
- La localisation de ces activités sur des éléments matériels

Un peu de terminologie

Architecture logicielle

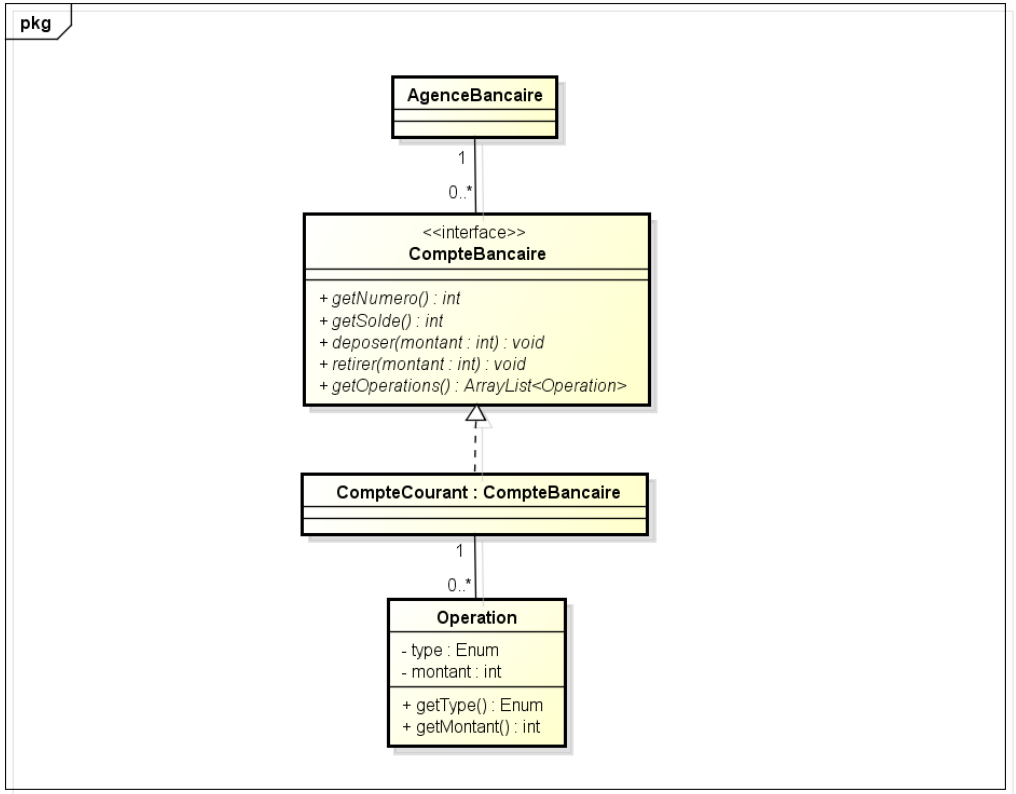
- Architecture statique
  - Description statique des éléments logiciels et de leurs relations
    - UML : diagrammes de classe UML, diagrammes de séquence*, diagrammes d'activité*, diagrammes d'état...

*Domaine des design patterns classiques

Un peu de terminologie

Architecture logicielle

- Architecture statique



Un peu de terminologie

Architecture logicielle :

- Architecture dynamique = description
  - des activités qui forment le système
    - Processus et threads
  - des relations entre ces activités
    - Création, synchronisation, échange de données et services, ...
    - En régime temporaire comme en régime établi
  - du mapping des éléments statiques sur ces activités

Architecture n-tiers : kesako ?

Une architecture n-tiers décompose un système

- en n **étages** (“tier” en anglais) ou **couches,**
- chaque étage réalisant un ensemble **homogène** de fonctions
- et communiquant avec les autres couches au travers d’une **interface bien définie**

Architecture n-tiers : kesako ?

Ensemble **homogène** de fonctions : fondamental !

- Spécialisation des différentes parties du système
  - IHM, base de données, algorithmes lourds et propres au domaine, ...
- On évite que chaque partie implémente des fonctions diverses
- On groupe les éléments d’une couche ensemble pour les livraisons
  - On parle de “packaging”

Architecture n-tiers : kesako ?

Ensemble **homogène** de fonctions : fondamental !

- Un développeur n’a pas besoin de connaître tous les détails de toutes les couches
  - Spécialisation
    - Développeur front-end
    - Développeur back-end
    - Développeur full-stack
    - Etc.

Architecture n-tiers : kesako ?

Interface bien définie entre couches

- Il n’y a pas différentes façons de s’interfacer entre couches
  - Risque de dépendances trop fortes et complexes entre couches
- Remplacement facile d’une implémentation de couche par une autre si l’interface reste la même
  - Exemple : JDBC

Architecture n-tiers : kesako ?

Interface bien définie entre couches

- Elle peut être programmatique (API = Application Programming Interface)
  - Entre deux composants logiciels dans le même espace d'adresse ==> appel de librairie

```
Connection conn = DriverManager.getConnection(DB_URL,USER,PASS);
Statement stmt = conn.createStatement();
String sql = "SELECT id, first, last, age FROM Employees";
ResultSet rs = stmt.executeQuery(sql);
while(rs.next()){
    int id  = rs.getInt("id");
```

Architecture n-tiers : kesako ?

Interface bien définie entre couches

- Elle peut être protocolaire
  - Typiquement entre deux composants distribués
    - ⇒ lecture / écriture sur socket
    - ⇒ dépôt / retrait sur file de messages
  - Exemples :
    - HTTP, entre un navigateur et un serveur Web
    - Java Message Service
    - Middleware ad-hoc

Architecture n-tiers : pourquoi ?

Pourquoi décomposer ?

- **Simplifier** la réalisation de chaque partie (“diviser pour régner”)
  - Spécialisation des traitements et donc du personnel par couche (moins de choses à connaître)
- Faciliter la **distribution** des traitements sur différentes machines
  - Répartition de la charge
  - Machines spécialisées

Architecture n-tiers

Exemples classiques de couches :

- **Présentation** ou **Client** (IHM, graphique ou pas)
  - Navigateur Web, application Android, ligne de commande...
- **Web** (gestion des requêtes et réponses HTTP)
- **Application** (fonctions applicatives fournies par ce système)
- **Métier** (concepts généraux au domaine d’application)
- **Données** ou **Persistance** (SGBDR souvent)

Architecture n-tiers

Différences entre couches Application et Métier

- **Métier** : concepts de base du domaine
- **Application** : façon dont les concepts sont organisés et mis en œuvre pour réaliser les fonctions de l'application
  - Les éléments de la couche Métier sont des briques sur lesquelles on bâtit les fonctions applicatives

Objets métier et logique applicative

Application 1 : gestion d’agence bancaire

- Gestion de comptes bancaires de différents types
- Concepts
  - Compte bancaire
  - Agence bancaire (ensemble de comptes bancaires)
  - Client
  - ...

Objets métier et logique applicative

Application 1 : gestion d’agence bancaire

- Opérations
  - Création / suppression de compte
  - Dépôt d’argent sur compte bancaire
  - Retrait d’argent sur compte bancaire
  - ...

Objets métier et logique applicative

Application 2 : société de Bourse

- Réalise des opérations boursières pour ses clients
- Concepts
  - Titre financier (action, obligation, option, ….)
  - Centre de cotation (Bourse de New York, ...)
  - Compte bancaire
  - Client
  - ...

Objets métier et logique applicative

Application 2 : société de Bourse

- Opérations
  - Informer les clients de la cotation des titres
  - Réaliser les achats / ventes pour les clients
  - Déposer / retirer l’argent des comptes bancaires
  - ...

Objets métier et logique applicative

Les 2 applications mettent en œuvre des concepts « métier » communs (compte bancaire, client, …) mais pour fournir des services différents

- La logique applicative est l’ensemble des fonctions fournies
- Les objets métier sont les éléments de base mis en œuvre par ces fonctions
  - Compte bancaire, client, ...

Objets métier et logique applicative

Cette distinction n’est pas toujours intéressante

- Petites applications
- Fonctions très spécifiques avec concepts métier uniques

Les objets métier sont souvent des objets persistants.

Architectures n-tiers

Architectures classiques

- Monolithique
  - Pas de séparation claire en couches, même machine
- 2 couches (souvent client / serveur)
  - Navigateur / serveur Web

    → Présentation / autres traitements
  - Application métier / SGBD

    → Présentation et traitements / stockage des données

Architectures n-tiers

Architectures classiques

- 3 couches (la plus classique)
  - IHM / traitements applicatifs / stockage des données
- 4 couches
  - IHM / couche Web / traitements applicatifs / stockage des données
    - Couche Web : gestion des requêtes et réponses HTTP → mise en relation de l'IHM et des traitements applicatifs

Architectures n-tiers

Architectures classiques

- 5 couches
  - IHM / couche Web / fonctions applicatives / objets métier / stockage des données

Le framework JEE

JEE (Java Enterprise Edition) est un **cadre architectural** (“**framework**” en anglais) bâti sur le langage Java et permettant de réaliser des applications n-tiers

Version actuelle = 8

Le framework JEE

Ce cadre définit :

- La décomposition en couches d'un système
- Les interfaces entre les couches
- Un modèle de **composant** logiciel
- Des conteneurs (**containers**) de composant qui fournissent des **services génériques** aux composants
- Un modèle de **serveur d'application** qui héberge la couche intermédiaire des applications (web / métier / applicatif)

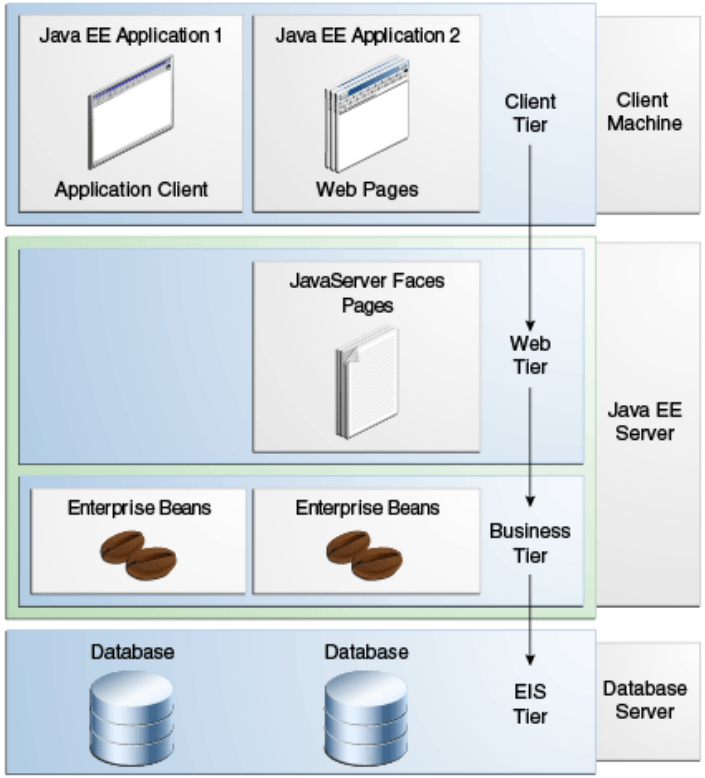
Les couches (« tiers ») du framework JEE

La décomposition en couches dans JEE

- Couche **Client** (genre navigateur Web ou IHM lourde)
- Couche **Business** (logique applicative et métier)
  - Décomposable en deux sous-couches
- Couche **Web** (si communication HTTP entre couches Client et Business)
- Couche **Data** (stockage des données) ou **EIS** (Entreprise Information System)

Les couches (« tiers ») du framework JEE

Exemple



La couche Business

Dans cette couche, JEE permet de distinguer la logique applicative et les objets Métier :

- EJB de type Session
- Entités (anciennement EJB de type Entity)

Il existe d’autres types d’EJB (Message-driven)

La couche Business

Avantages des EJB (composants Business) :

- Les services communs (sécurité, transactions, accès aux sources de données…) sont fournis par le container d'EJB, donc les composants EJB n'ont pas à s'en occuper
- Ils implémentent la logique applicative, et donc rendent les applications clientes plus légères
- Ce sont de vrais composants, qui peuvent réutilisés pour former de nouvelles applications

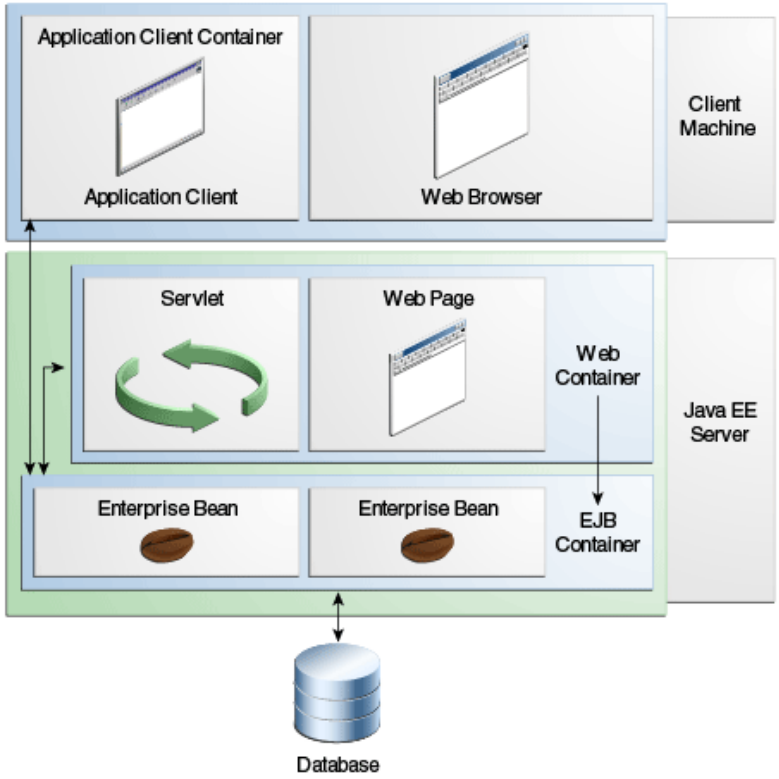
Les containers JEE

Les containers sont l'interface entre les composants et le serveur d'application qui fournit en pratique les services génériques :

- Transactions
- Sécurité (accès aux ressources selon les droits des utilisateurs)
- Annuaire (accès à des ressources par leur nom)
- Connexions distantes
- Cycle de vie des composants (servlets, EJB, ...)
- Messages

Les containers JEE

Exemple



Les containers JEE

Types de containers :

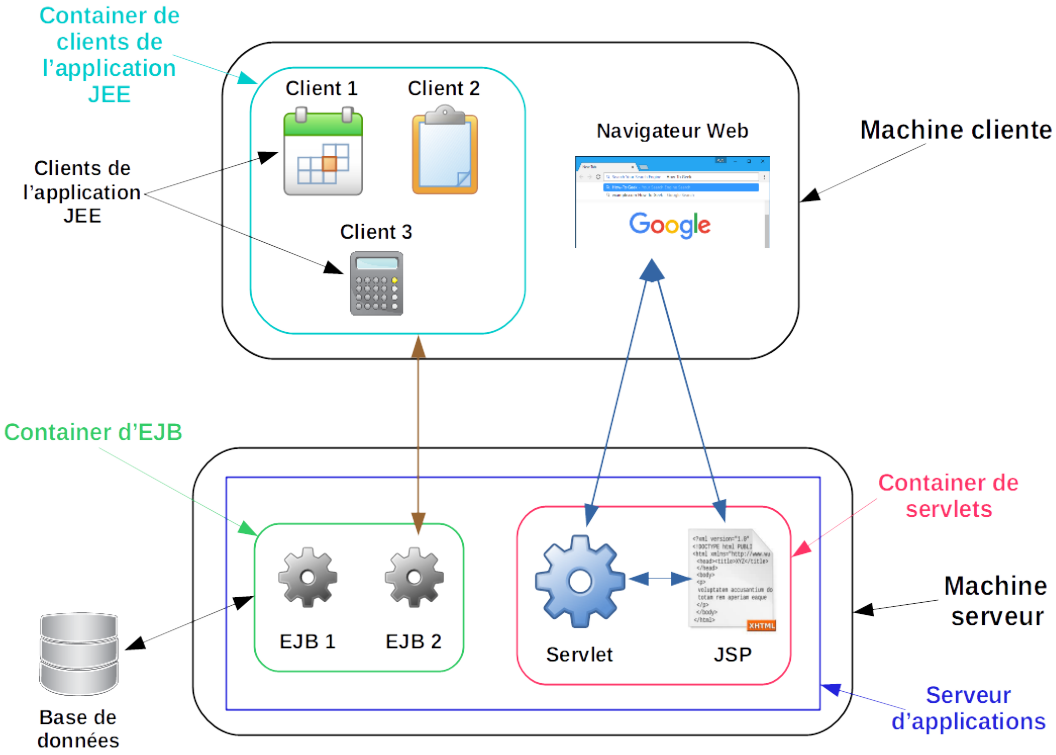
- Container Web ⇒ couche web
  - Tomcat et Jetty sont des containers standalone
- Container EJB ⇒ couche métier + applicative
- Application client container ⇒ couche Présentation
- Applet container (pour historique)

Les composants JEE

Exemples de composants :

- Servlets
- Java Server Pages (JSP)
- Java Server Faces
- Entreprise Java Beans (EJB)

Les éléments du framework JEE



L’assemblage de composants JEE

Les composants sont assemblés en **modules**, ensemble de composants du même type :

- modules EJB
- modules Web
- modules « client d’application »
- modules d’adaptation de ressources (partie EIS)

L’assemblage de composants JEE

Un module peut contenir un ou plusieurs fichiers de configuration appelés **deployment descriptors** :

- adapte le fonctionnement de l’application à un contexte particulier
- évite l’adaptation par modification du code source

Certains fichiers sont communs à tous les serveurs d’application JEE, d’autres sont spécifiques.

La couche Business

La couche Business (couche Métier) inclut tous les composants implémentant la **logique applicative** et l'**accès aux données**.

En gros :

- Objets métier ⇒ entités persistentes
- Logique applicative ⇒ EJB Session et "Message-driven"