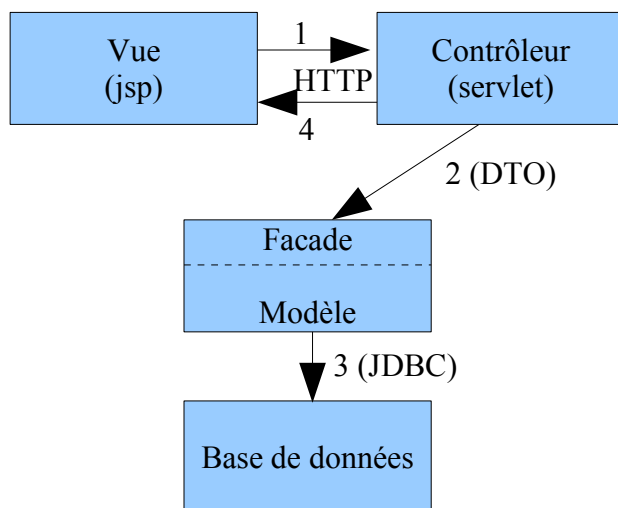


## Leçon 0 – Introduction

*Dans cette leçon, nous revoyons ce que nous avons fait en 2ème année et nous présentons les technologies que nous verrons cette année.*

### 1 RAPPELS DU COURS DE L'ANNÉE PASSÉE

Voici un schéma de l'architecture d'une application web comme nous les avons construites l'année passée.



1. Une vue est présentée au navigateur. L'utilisateur effectue une action. Cette action est transmise au contrôleur (unique ou un par rôle ou un par sous-système)
2. Le contrôleur analyse la requête de l'utilisateur et y répond. Il utilise pour cela le modèle. Afin de faciliter l'utilisation du modèle, celui-ci propose une (ou plusieurs) façade(s). Lorsque de l'information doit être échangée avec le modèle, cela se fait au travers de DTO (Data Transfer Object)
3. Le modèle utilise JDBC pour accéder à la base de données et répondre ainsi à la demande initiée par le contrôleur. (lecture ou modification des données). Le modèle offre en quelques sortes une vue OO de la BD relationnelle.
4. Le contrôleur choisit la vue suivante à présenter au navigateur. La vue utilise les attributs (de requête, de session) pour construire le texte précis (un document HTML). Ses attributs peuvent être des DTO.

### Exercice

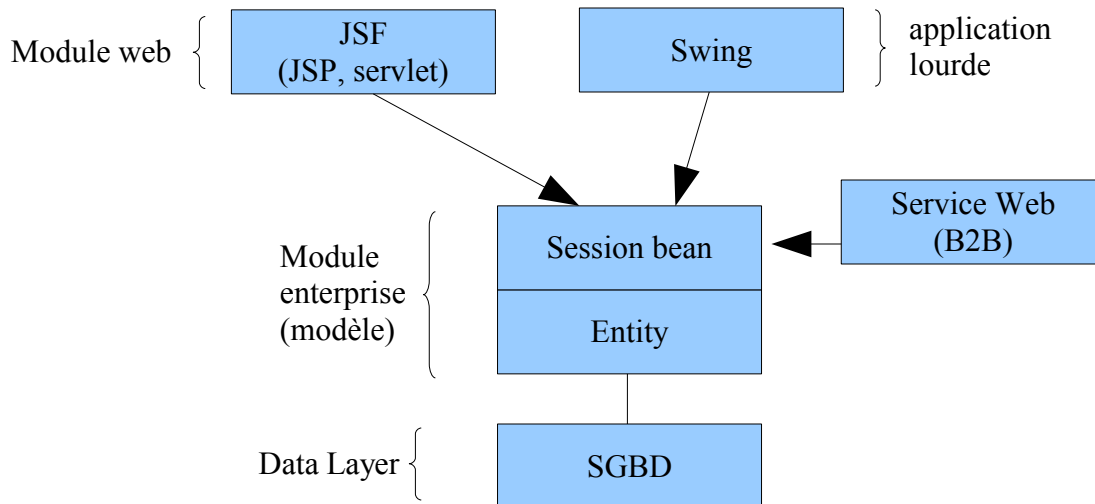
Refaites un petit exercice pour vous rafraîchir la mémoire.

1. Vérifiez l'installation des outils; il faudra probablement installer le Sun Application Server.
2. Reprenez le modèle de la bibliothèque de l'année passée. Si vous ne l'avez plus, on vous fournit une base de départ.
3. Reprenez l'application web faite l'année passée (avec un menu)
4. Ajoutez une entrée de menu pour obtenir les informations détaillées sur un lecteur (nom, prénom, nombre maximum d'emprunts et liste des ouvrages empruntés).
5. Portez un regard critique sur ce que vous venez de produire.

## 2 CE QUE NOUS VERRONS CETTE ANNÉE

---

Le schéma vu l'année passée est fortement simplifié. De nombreuses autres technologies sont mises en oeuvre dans les applications distribuées. Il est temps de les voir (les principales en tout cas). Voici un petit schéma qui les reprend et montre comment elles s'articulent.



- Les **entités** s'occupent de la persistance des données. Ils permettent de manipuler des objets qui sont en lien étroit avec des tables de DB. Ils assurent ainsi le mapping OO-relationnel.
- Les **beans de sessions** permettent de coder la logique métier et offrent de nombreuses facilités (comme une gestion des transactions)
- Pour le module web nous utiliserons **JSF** qui permet d'automatiser la tâche du contrôleur, d'offrir des interfaces web plus riche et de configurer facilement la transition entre les pages. Nous auront l'occasion d'aborder l'**expression langage** et les **tags core** qui facilitent l'écriture des JSP et augmentent leur lisibilité.
- Nous verrons également comment accéder directement au sessions beans à partir d'une **client riche** (en Swing par exemple).
- Les **services web** sont assez nouveaux et sont dédiés aux B2B. Ils permettent d'échanger des données entre des applications distribuées hétérogènes.
- Nous verrons ce que Java EE 5 offre pour gérer la problématique des **transactions**, de **l'authentification** et de la **sécurité**
- Si il nous reste du temps, nous verrons les **message-driven beans** qui permettent le travail asynchrone.

## 3 LES OUTILS

---

### Java SE

Java en est à sa version 6 (installée à l'école). La version 5 convient également. Depuis cette version 5, le langage s'est étoffé. Cette année nous ferons un usage important des **annotations**. Il est utile de vous familiariser avec ce concept, par exemple avec ce tutoriel :

<http://java.sun.com/docs/books/tutorial/java/javaOO/annotations.html>

### Netbeans

C'est l'IDE que nous allons utiliser (comme l'année passée). Il en est à la version 5.5.1.

### Java EE 5

Java Enterprise Edition en est à la version 5 (ne plus utiliser la version 4). Notez aussi que le 2 a disparu (on est passé de J2EE4 à Java EE 5). Java EE 5 n'est qu'une spécification. Plusieurs serveurs d'applications l'implémentent.

### Le Sun Application Server

Une implémentation de Java EE, développé par SUN. On en est à la version 9. Ne plus utiliser la 8 qui n'implémente pas la version 5 de Java EE. Sun vient de rendre public ce produit qui porte le nom de « Glassfish » dans sa version OpenSource. A ma connaissance, il n'y a pas de différence fonctionnelle entre le « Sun Application Server » et « Glassfish ».

### Derby

Le serveur d'application de Sun offre en standard Derby, un SGBD. Si vous travaillez sur un environnement hétérogène (par exemple Windows à l'école et Linux à la maison) il vous permet d'avoir un même SGBD ce qui simplifie le portage. De plus, il est très bien intégré à Netbeans.