Module JEE/Spring

Vincent THOMASSIN

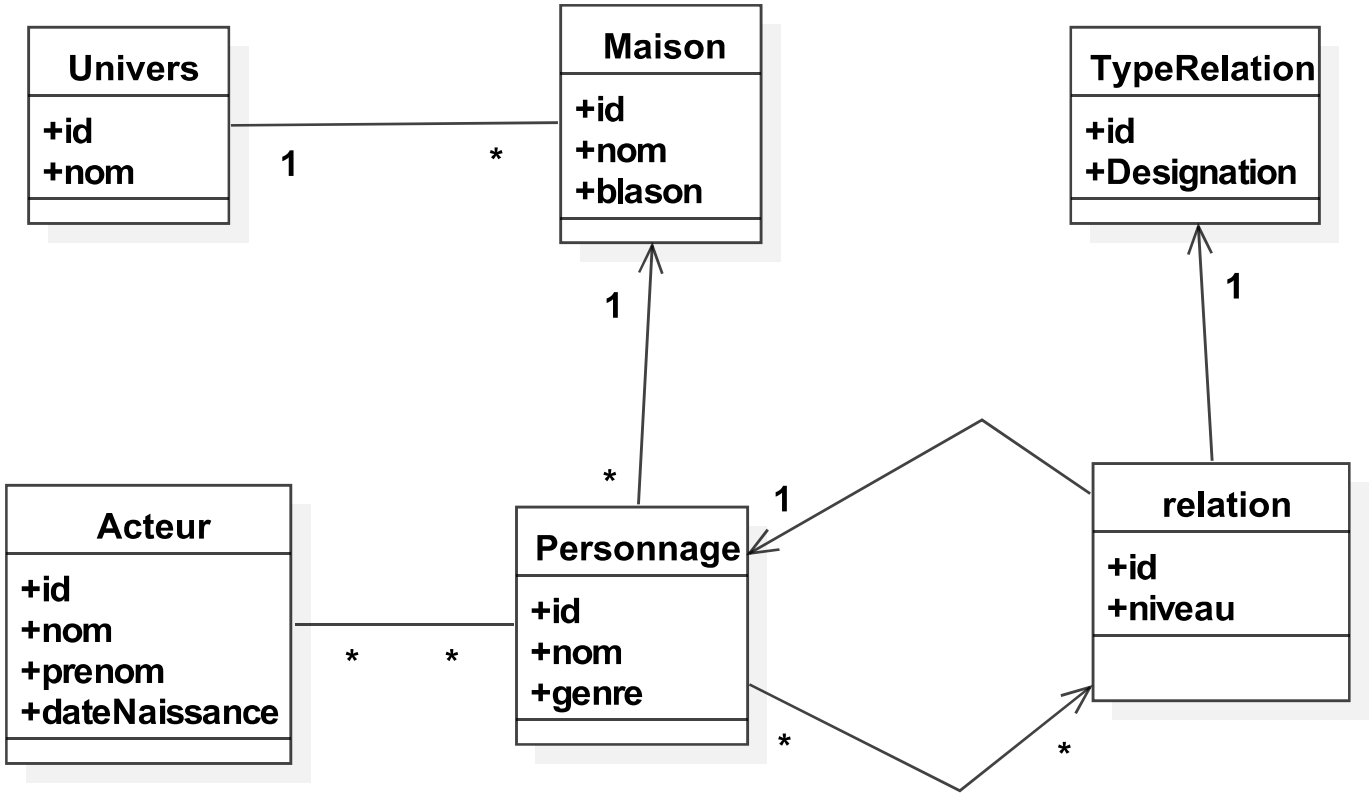
Au programme

Réaliser une application Java EE grâce aux Framework Hibernate et Spring

- Introduction au Framework
- L'inversion de contrôle
- L'intégration avec Hibernate
- fondamentaux HTTP/REST
- Premier webservice
- Principe d'industrialisation des développements : (Maven, Junit)
- Serveur d'application - Tomcat

Fil Rouge

Construction du Backend exposant l'API des personnage défini lors des cours précedents



Evolution des approches de programmation

Programmation **Procédurale**	• Sous programmes: Procédures, fonctions (Basic, Pascal, C, Fortran,..)
Programmation Orientée **Objet**	• Objet = Etat+ Comportement + Identité • Concepts fondamentaux : Objet, classe, Héritage, polymorphisme, encapsulation (C++, JAVA, C#, ..)
Programmation Orientée **composants**	• Objets distribués, réutilisables, configurables, Interchangeables, évolutifs, mobiles, surveillable : Containers (EJB, Spring) : AOP
Programmation Orientée **Services**	• Composant disponibles à d'autres applications distantes hétérogènes via des protocoles (http) transportant des données: XML, JSON => (SOAP+WSDL+UDDI) et REST (HTTP+JSON)

Exigences d'un projet informatique

Exigences fonctionnelles:

Une application est créée pour répondre , tout d'abord, aux besoins fonctionnels des entreprises.

Exigences Techniques :

Les performances:

- Montée en charge
- Équilibrage de charges
- Haute disponibilité et tolérance aux pannes

La maintenance:

- Faire évoluer les applications d'une manière très simple

Sécurité

Transaction

Portabilité

Distribution

Journalisation

Exigences financières :

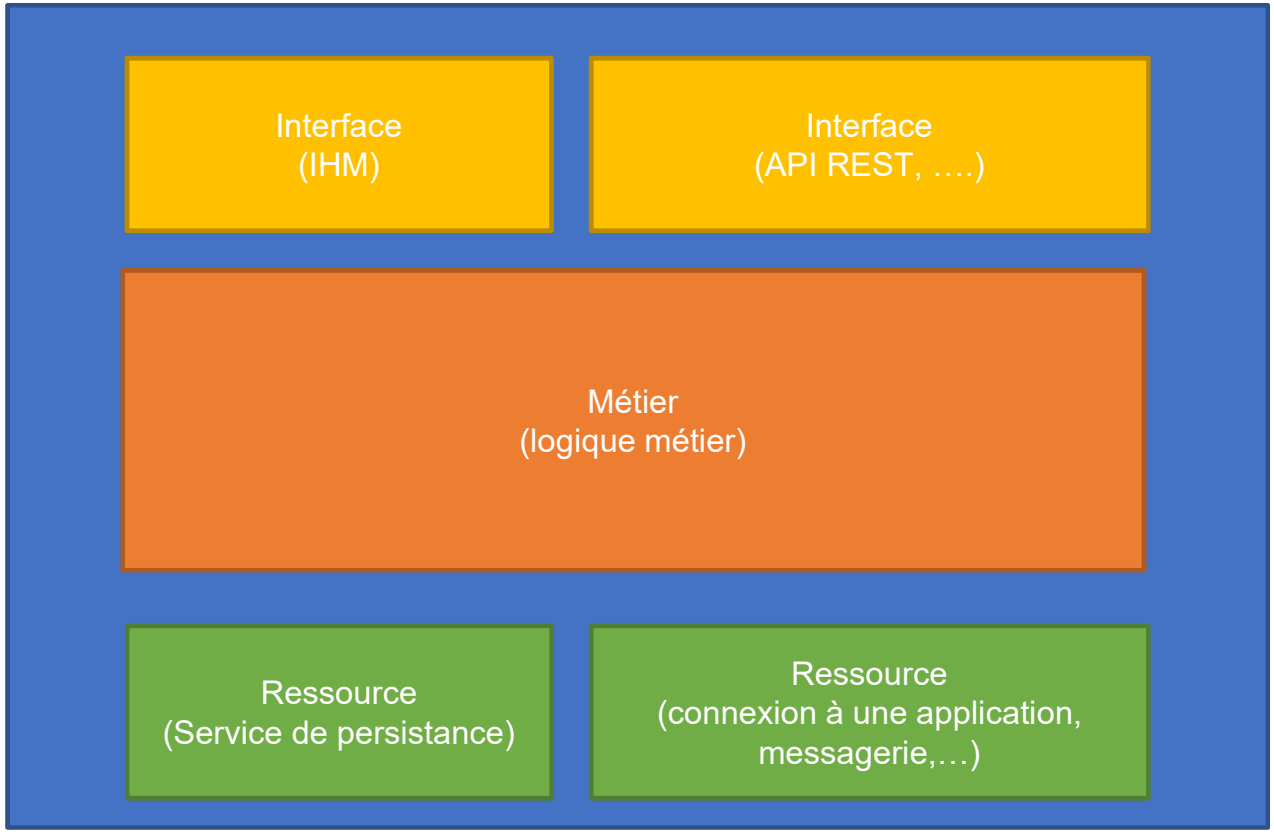
Coût du logiciel

Pourquoi un Framework ?

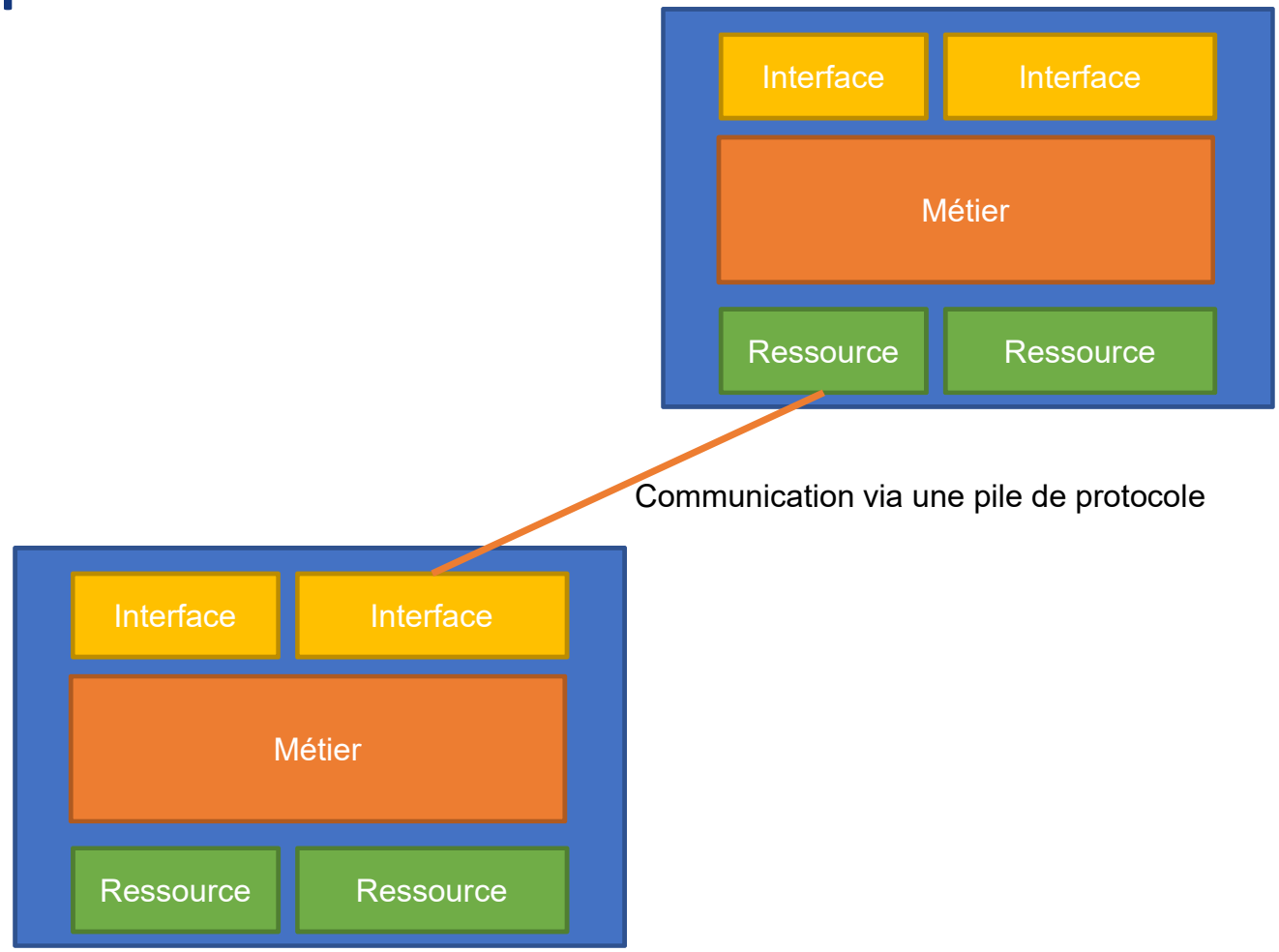
Il est très difficile de développer un système logiciel qui respecte ces exigences sans utiliser l'expérience des autres :

- Le Framework permet au **développeur** **de se concentrer sur le code métier (Exigences fonctionnelles)**
- **Le Framework s'occupe du code Technique (Exigence Technique)**

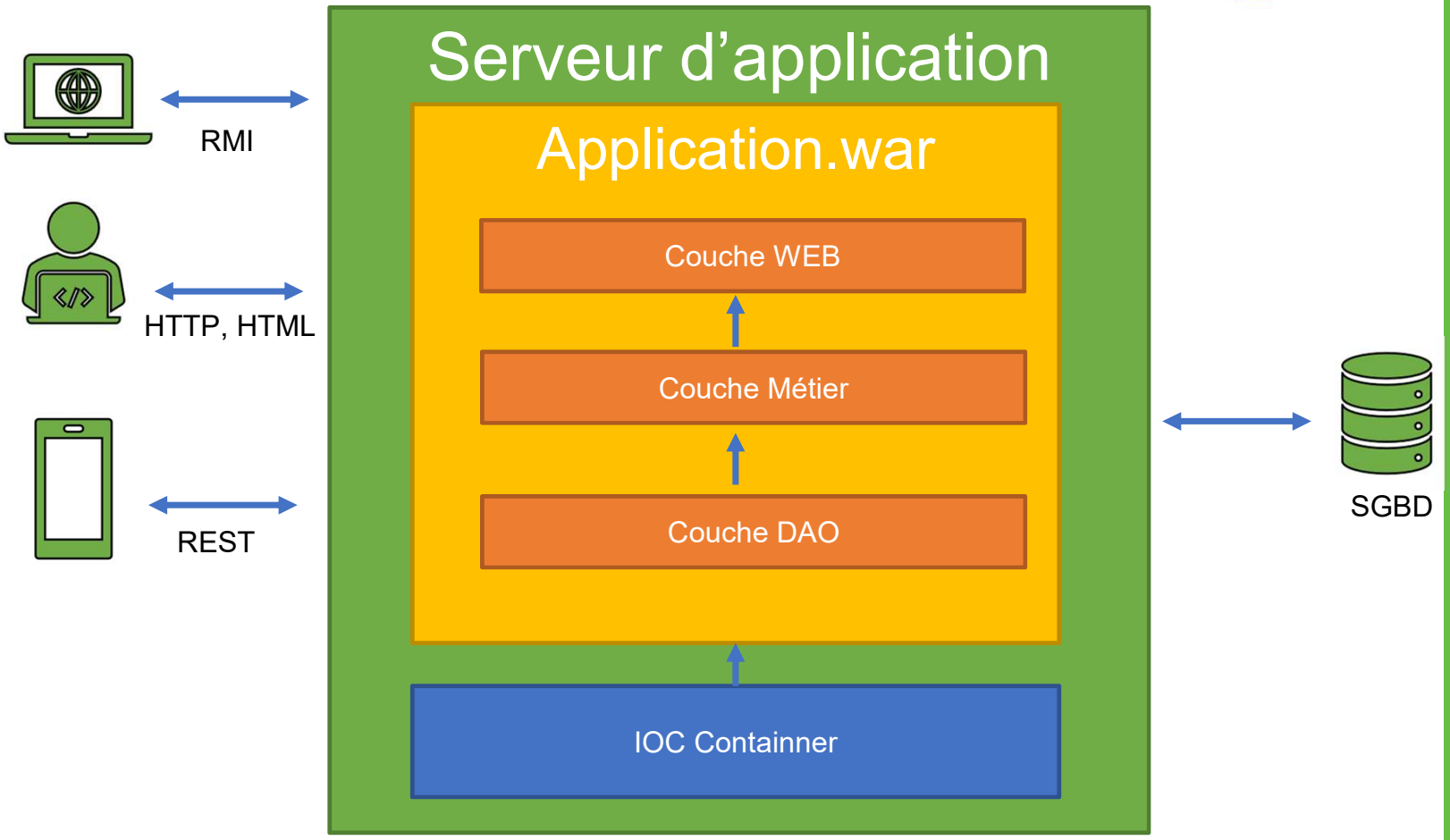
Application N-Tiers



Application N-Tiers



SPRING Framwork



SPRING Framework

Grands principes :

- ## L’inversion de contrôle

C’est le Framework qui fait appel à la applicatif est pas l’inverse.
Le Framework gère le cycle de vie des composants en fonction de la configuration.

- ## Convention over Configuration

On ne configure que ce qui est spécifique à notre application le reste est laisser dans l’état de l’art proposer par le Framework.

API REST

Les API REST sont basées sur HTTP. Http comme les pages web. C'est un protocole qui est donc en général ouvert sur les reseaux.

Vous les avez consommé lors du module HTML avec la base des personnages.

Les réponses du serveur pour les API REST peuvent être délivrées dans de multiples formats. JSON (JavaScript Object Notation) est souvent utilisé

Voir : https://openclassrooms.com/fr/courses/3449001-utilisez-des-api-rest-dans-vos-projets-web

API REST : Les ressources

Comme une API REST est un système d'URI, les ressources déterminent la structure des URI

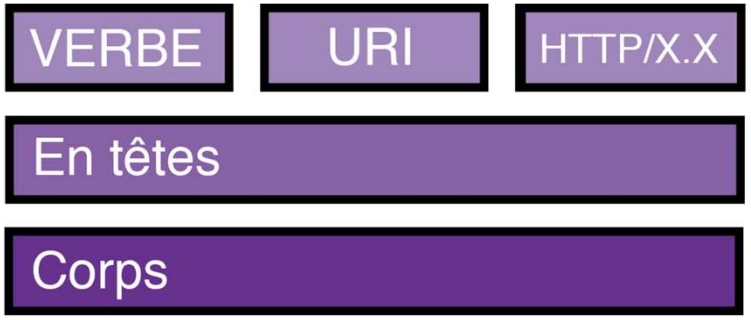
Une ressource est un objet ou plusieurs objets auquel les utilisateurs de votre API peuvent vouloir accéder.

Exemples de désignation de resources :
La liste de utilisateurs :
`/users`
Un utilisateur en particuliers
`/users/238`
Les consignes de livraison pour une adresse
`/addresses/195/delivery_notes`

Note : les ressources peuvent également désignées des opérations « métier ».

API REST : Les requêtes

Défini par le protocole HTTP, ces informations sont envoyées au serveur,



Le verbe peut être GET, PUT, POST, DELETE et encore plus

Le corps peut contenir de la données utiles et/ou bien des paramètre

Voir : https://developer.mozilla.org/fr/docs/Web/HTTP/M%C3%A9thode/POST

Comme vu précédemment l'URI désigne la ressources, elle peut également être compléter par des paramètres.

API REST : Les méthodes

Les méthodes REST sont les verbes HTTP

GET

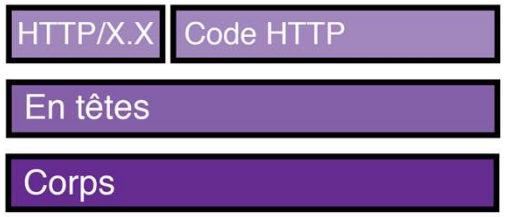
GET est la méthode la **plus utilisée** pour les requêtes HTTP. Cette méthode existe pour récupérer des données d'une ressource.

```
GET http://site.com/users?parametre=exemple&autre_parametre=exemple2
```

vous pouvez envoyer des paramètres à l'URI en forme d'un query string pour pouvoir recevoir les données spécifiques

Ne jamais faire de modifications coté serveur suite à une sollicitation GET.
Les navigateurs permettent à n'import quel site de transmettre des requêtes sur n'importe quel autre serveur.
Ce n'est pas le cas pour les autres verbes utilisés par REST.
Voir OWASP pour plus de recommandations: https://owasp.org/www-project-top-ten/

API REST : Les réponses



code réponse HTTP
Un code de 3 chiffres qui indique l'état de la réponse
2xx indique le succès.
3xx redirige le client ailleurs.
4xx indique une faute de la part du client.
5xx indique une erreur de la part du serveur.

Le **corps de la réponse** contient des données pour les utilisateurs des API REST. Quand vous regardez un site web sur votre ordinateur, le corps d'une réponse est normalement en HTML pour que vous puissiez voir le site mis en forme.
Par contre, dans une situation API, quand deux applications communiquent entre elles, le corps de la réponse est souvent en forme de JSON

API REST : Les méthodes

Les méthodes REST sont les verbes HTTP

POST
Envoyer des données / ajouter une ressources

PUT
Modification d'une ressources

DELETE
Effacement d'une ressources

Exemple :

```
POST /users HTTP/1.1
name=jessica
```

Reference

http://www.youssfi.net/
https://openclassrooms.com/fr/courses/3449001-utilisez-des-api-rest-dans-vos-projets-web
https://owasp.org/www-project-top-ten/

A vous de Jouer !