Iterations-rapports : Récapitulatif

Fichier téléchargeable à l'adresse :
http://axel.ammuller.free.fr/ezbay/iterations_rapport_recapitulatif.pdf

ITERATION 00

[T03] Méthodologies Agiles / J2EE et webServices

Projet ezBay

Rapport d'avancement du prototype technique

1. Constitution du groupe

Axel AMMÜLLER (axel.ammuller@free.fr)
Lotfi ELGHERBI (lotfi_elgh@yahoo.fr)
Mohamed EL-MRABH (elmrabah@yahoo.fr)
Sophie LION (lion.sophie@free.fr)
Nous avons créé un compte mail pour le groupe, vous pouvez nous joindre à l'adresse a***xlomoso@gmail.com***

2. Installation de l'environnement de travail:

2.1. Pour l'aspect développement

Nous avons choisi les logiciels suivants :

- Eclipse 3.1.1
- plugin MyEclipse 4.0.3
- JBoss 4.0.3
- MySql Server 5.0 (ainsi que les outils d'administration mysql_administrator et mysql_Query_browser)

2.2. Pour l'aspect collaboratif

Pour les outils de travail en groupe, nous avons choisi d'utiliser les services du site Internet www.berlios.de. Celui-ci est une plate-forme de travail pour projets open source. Il donne accès à un certain nombre d'outils comme :

- L'accès à un Repository CVS pour le projet
- Un Serveur MySql
- Divers outils de travail collaboratif tels que forums de discussion, outil de gestion des tâches, outil de gestion des bugs…

Chacun des membres du groupe possède maintenant le même environnement de travail, avec les mêmes configurations.

2.3. Premiers tests de développement

Afin de valider notre plate-forme technique, nous avons effectué un premier test de développement d'un *EJB Session Stateless « HelloEJB »*, possédant une méthode `sayHello()` retournant simplement un objet de type `String`. Un programme client java standalone appelle l'interface locale du Bean et invoque sa méthode `sayHello()`. Le résultat est affiché dans la console de l'IDE.

3. Difficultés rencontrées

3.1. Choix de la plateforme technique :

La principale difficulté a été de trouver (et de choisir !) les différents logiciels et plugins permettant d'effectuer nos développements dans un environnement intégré. C'est pour cela que nous avons choisi d'utiliser MyEclipse, qui possède par défaut certains plugins très utiles (xDoclet pour générer les différents fichiers liés aux Beans, plugins de lancement de JBoss et de déploiement des EJB).

3.2. Développement du premier Bean :

Pour ce premier test, nous nous sommes inspirés de différents tutoriaux trouvés sur Internet. C'est cette partie qui a été la plus laborieuse. La création de l'EJB ainsi que son déploiement ont été choses faciles. Par contre, l'appel par un programme client nous a posé des problèmes, pas encore tout à fait résolus pour l'instant. Par exemple, concernant la récupération de l'interface du Bean, grâce à JNDI !! Parfois le service en question ne semble pas joignable sur toutes nos machines…

ITERATION 01

[T03] Méthodologies Agiles / J2EE et webServices

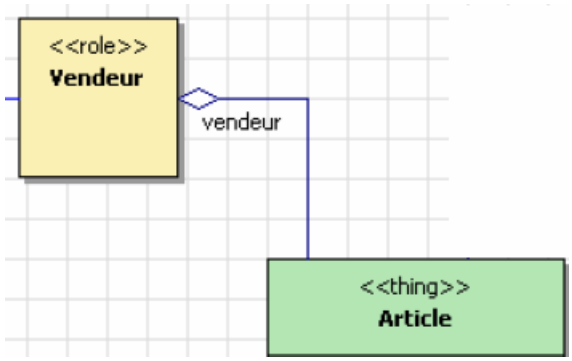
Projet ezBay

Prototype technique

4. Développement

4.1. Partie Serveur

Nous nous sommes concentrés sur les classes Vendeur et Article.



- 2 CMP Entity Bean :

  - ArticleBean

    Attributs :

```
private String id;
private String libelle;
private String marque;
private String modele;
private Double prixVente;
private Integer anneeFabrication;
private java.lang.String description;
private Date dateLimite;
```

    Méthodes :

```
Accesseurs en lecture et écriture
public abstract VendeurLocal getVendeurLocal();
public abstract void setVendeurLocal(VendeurLocal vendeurLocal);
public abstract ArticleDTO getArticleDTO();
public void updateArticle(ArticleDTO articleDTO)
```

    Finder (EJB-QL) :

```
java.util.Collection findAll()
java.util.Collection findByLibelle(java.lang.String libelle)
java.util.Collection findByMarque(java.lang.String marque)
java.util.Collection findByModele(java.lang.String modele)
java.util.Collection findByPrixVente(java.lang.Double prixVente)
java.util.Collection findByAnneeFabrication(java.lang.Integer anneeFabrication)
java.util.Collection findByDateLimite(java.util.Date dateLimite)
java.util.Collection findByDescription(java.lang.String description)
```

  - VendeurBean

    Attributs :

```
private String id;
private String nom;
```

    Méthodes :

```
Accesseurs en lecture et écriture
public abstract Collection getArticle();
public abstract void setArticle(Collection article);
public abstract VendeurDTO getVendeurDTO();
public void updateVendeur(VendeurDTO vendeurDTO) ;
```

    Finder (EJB-QL) :

```
java.util.Collection findAll()
java.util.Collection findByNom(java.lang.String nom)
```

Création des DTO : ArticleDTO et VendeurDTO

- 2 EJB Stateless Session Facade :
  - ArticleFacadeBean
  - VendeurFacadeBean

- 1 relation CMR entre les EJB Entity
  - Article-Vendeur (bidirectionnelle) : à un VendeurBean sont associés plusieurs ArticleBean.

4.2. Partie Client

Des recherches sur Internet (et en toute franchise, des conseils avisés d'un autre groupe), nous ont permis de comprendre que l'API JUnit n'était pas utilisable dans le cadre des EJB, c'est pourquoi nous avons utilisé JUnitEE pour les tests de nos Beans à travers les interfaces locales (pour les Entity), et les interfaces distantes (pour les Session).

Exemple de rapport de tests

JUnit Test Results

Summary of test results

✓	ℹ	ezbay.test.ArticleFacadeTest	3,047 sec

List of executed tests

ezbay.test.ArticleFacadeTest		
✓	testGetArticle(ezbay.test.ArticleFacadeTest)	1,297 sec
✓	testGetVendeurDTO(ezbay.test.ArticleFacadeTest)	0,265 sec
✓	testGetArticlesByLibelle(ezbay.test.ArticleFacadeTest)	0,313 sec
✓	testUpdateArticle(ezbay.test.ArticleFacadeTest)	1,172 sec

ArticleFacadeTest.java et VendeurFacadeTest.java qui comme leur nom l'indique sont des classes de tests pour Article et Vendeur.

5. Difficultés rencontrées

La construction de la liaison CMR entre Article et Vendeur nous a posé quelques problèmes, pour les raisons suivantes :

- difficultés de compréhension de la gestion CMR par le « container » : Que fait-il ? Que ne fait-il pas ?...

- méthode de programmation : l'utilisation des tags xDoclet ainsi que la structure imposée par les EJB imposent une implémentation assez lointaine de la programmation Java telle que nous la connaissions auparavant.