Diagramme de classe effectif

Problèmes rencontrés

1. GUI
2. BL
3. Persistance

Gestion de projet

1. Répartition générale
2. Gestionnaire de tâches
3. Gestionnaire de versions
4. Outils de conception
5. Outils de développement
6. Outils de tests

# 2. Problèmes rencontrés

## 2.3. Persistance

L’une des difficultés que nous avons rencontrées vis-à-vis de la persistance des données a été de trouver le driver nécessaire à la connexion à la base de données du logiciel. Il a également fallu gérer le fait que la base de données utilisée était commune afin d’éviter des modifications chaotiques. Pour cela nous avons donc décidé de copier la base de données communes sur une base de données qui nous était privée afin d’être libre de faire les modifications qu’il nous plaisait lors de nos tests.

Les vues ont été créées et sont fonctionnelles si on utilise SQL\*Plus, en revanche il nous a été impossible d’afficher les types nested table sur SQLDeveloper et en Java. Nous pensons que le problème est dû au descripteur du type qui n’a pas été construit : *Echec de construction du descripteur: Unable to resolve type "THIBAUT.ROUQUETTE.NT\_ENSEIGNEMENT"*

# 3. Gestion de projet

### 3.3. Gestionnaire de versions

Il nous a semblé important de mettre en place des outils de gestion permettant de partager le code de manière simple, de s’assurer que les différentes versions les plus à jour sont centralisées et de faciliter l’accès aux dernières versions du code pour d’éventuelles modifications.

Nous avons ainsi décidé de mettre en place un système de gestion de versions. Notre choix s’est tourné vers le logiciel client TortoiseSVN car il est libre et propose une interface intuitive. Grâce à son intégration dans l’explorateur Windows, un simple clic droit dans un dossier permet d’accéder à un menu permettant de mettre à jour la version sauvegardée localement et de soumettre la nouvelle version par rapport aux modifications effectuées en local.

Pour ce qui est du serveur de stockage des fichiers, nous nous sommes tournés vers Google Code, notamment car le stockage est gratuit, nous le connaissions déjà et nous utilisions également Google Document pour stocker les premiers rapports.

### 3.4. Outils de conception

Par rapport aux outils de conception, nous nous sommes d’abord orientés vers LucidChart, qui est un site d’édition de diagrammes collaboratif en ligne supportant le langage UML. Nous nous sommes orientés vers cette solution car elle proposait des outils collaboratifs en ligne et un stockage partagé des fichiers.

Cependant, après avoir commencé à faire les analyses pour la conception et en réalisant le nombre de classes que l’on avait, il nous a paru important de disposer d’un outil capable de générer du code automatiquement en fonction des diagrammes. C’est pourquoi nous nous sommes alors tournés vers BoUML, qui est un logiciel gratuit d’édition de diagrammes, qui est capable de générer du code. De plus, BoUML présentait l’avantage d’avoir des outils de gestions de classes déjà définies, nous permettant de réutiliser du travail déjà effectué, alors que réutiliser des classes dans LucidChart impliquait de les redessiner. BoUML n’est pas un outil collaboratif, mais la mise en place du gestionnaire de version nous a permis de palier à ce problème en incluant les fichiers de conception du logiciel dans les dossiers partagés.

Pour gérer les différents fichiers BoUML, nous avons d’abord fait deux fichiers séparés concernant chacun deux Use Case. Nous avons ensuite fusionné ces deux fichiers, puis traduit tous les noms en anglais car nous avions défini que la langue utilisée pour le codage serait l’anglais.

### 3.5. Outil de développement

Concernant l'outil pour développer les fonctionnalités que nous avons définies, nous avons choisi de nous tourner vers le logiciel Eclipse. Eclipse est un IDE très puissant et réputé être le meilleur pour ce qui est de la conception en Java. Il propose plusieurs fonctionnalités très intéressantes, comme de l’auto-complétion, une liste d'attributs et de méthodes de la classe actuelle, ainsi qu'une console intégrée. Il propose, en outre, une compilation à la volée, ce qui permet de voir très rapidement les erreurs de codage, ou encore, en laissant la souris sur une fonction, il nous en affiche la Javadoc. Une autre des raisons pour lesquelles nous nous sommes tournés vers cet IDE est que nous avions déjà eu l'occasion de nous en servir lors d'autres projets et nous le connaissons donc assez bien.