Binôme :
Pierrick CHOVELON
Julien TIRON

FIP 1A

Rapport Projet ToutAvis
INF112

Formation d'ingénieur par partenariat

Version : 1

26 mai 2015

TELECOM Bretagne

Table des matières

Introduction

Ce rapport a pour but de dresser un bilan le plus complet possible du projet fil rouge mené en INF112. Ce travail en binôme a pour but de nous faire découvrir les étapes d'un projet de développement logiciel de grande ampleur. Vous trouverez dans ce rapport les différentes informations concernant le projet ToutAvis et le travail que nous avons fourni pour le mener à bien. Vous trouverez dans chaque section quelques explications permettant de mieux comprendre notre démarche.

Le rapport présente tout d'abord l'investissement en temps dans le projet ainsi que les chiffres représentatifs du développement. Ensuite, nous présenterons la structure de notre logiciel grâce aux diagrammes de classes et de séquences. Enfin, nous dresserons le bilan du projet grâce aux différents rapports d'audit et de recette.

Fiche tâches/temps

Séance	Temps passé en P.H	Temps passé hors séance en P.H	Détails de l'activités
BE1	2x3h	0	Rédaction du CdCF
BE2	2x3h	2x2h	Rédaction de la stratégie de validation, de la fiche de tests et codage des tests
BE3	2x3h	2x1h	Élaboration diagrammes
TP1	2x3h	2x1h	Javadoc et début codage
TP2	2x3h	2x2h + 1h	Codage des méthodes et des tests
TP3	2x3h	2x4h	Ajout de la fonctionnalité «noter un avi»
TP4	2x3h	2x3h	Mise à jour du code de la moyenne en prenant en compte le karma du membre
TP5	2x3h	2x2h	Modification de code selon les remarques du groupe qui à fait la relecture
TP6	2x3h	2x1h + 3h	Modifications minimes du code et ajout information dans la javadoc
Total : $90P.H$	54	36	

- *Productivité en lignes de code source :*

La métrique de notre projet remonte que nous avons écrit 945 lignes de code pour le package *avis* (sans les tests ni les commentaires). En estimant avoir travaillé 90 heures, nous arrivons à $\frac{945}{90} = 10.5$ lignes de codes par heure pour deux personnes. Ainsi un membre du binôme a écrit 5 lignes par heure de travail.

- *Résultat financier de l'exercice :*

En partant du principe que le cout de revient d'une heure de travail d'un ingénieur junior s'élève à 60€ et que le projet a nécessité 90 heures de travail, nous chiffrons le coût de notre activité à 5400€.

- *Coût de la facturation du projet au client :*

Diagramme de classes et diagrammes de séquences

Vous trouverez dans les pages suivantes, les 3 diagrammes qui nous ont été demandés.

Nous avons essayé de faire un diagramme de classe le plus simple possible pour facilité la compréhension de la structure du projet pour des personnes extérieur à celui-ci. Certaines méthodes déclarées en *private* n'apparaissent pas ici (caprises de eUML). Elles se trouvent néanmoins dans la javadoc et dans les annexes.

Le diagramme de séquence de reviewOpinion (Fig. 2) peut sembler trop simple. Nous avons choisi de ne représenter que le cas de fonctionnement normal (i.e les données passées en paramêtres sont valides).

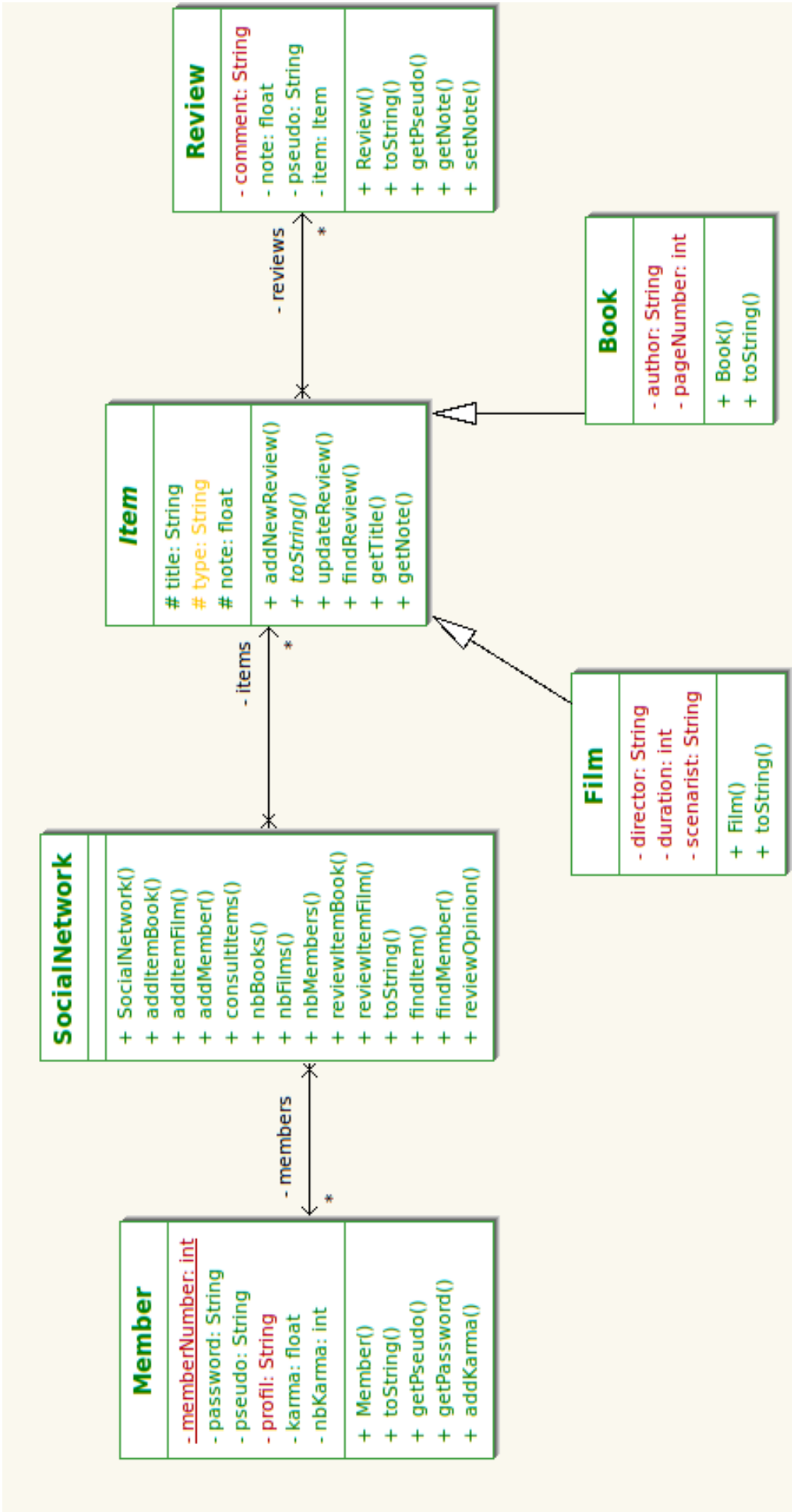

Figure 1 – Diagramme général de classes

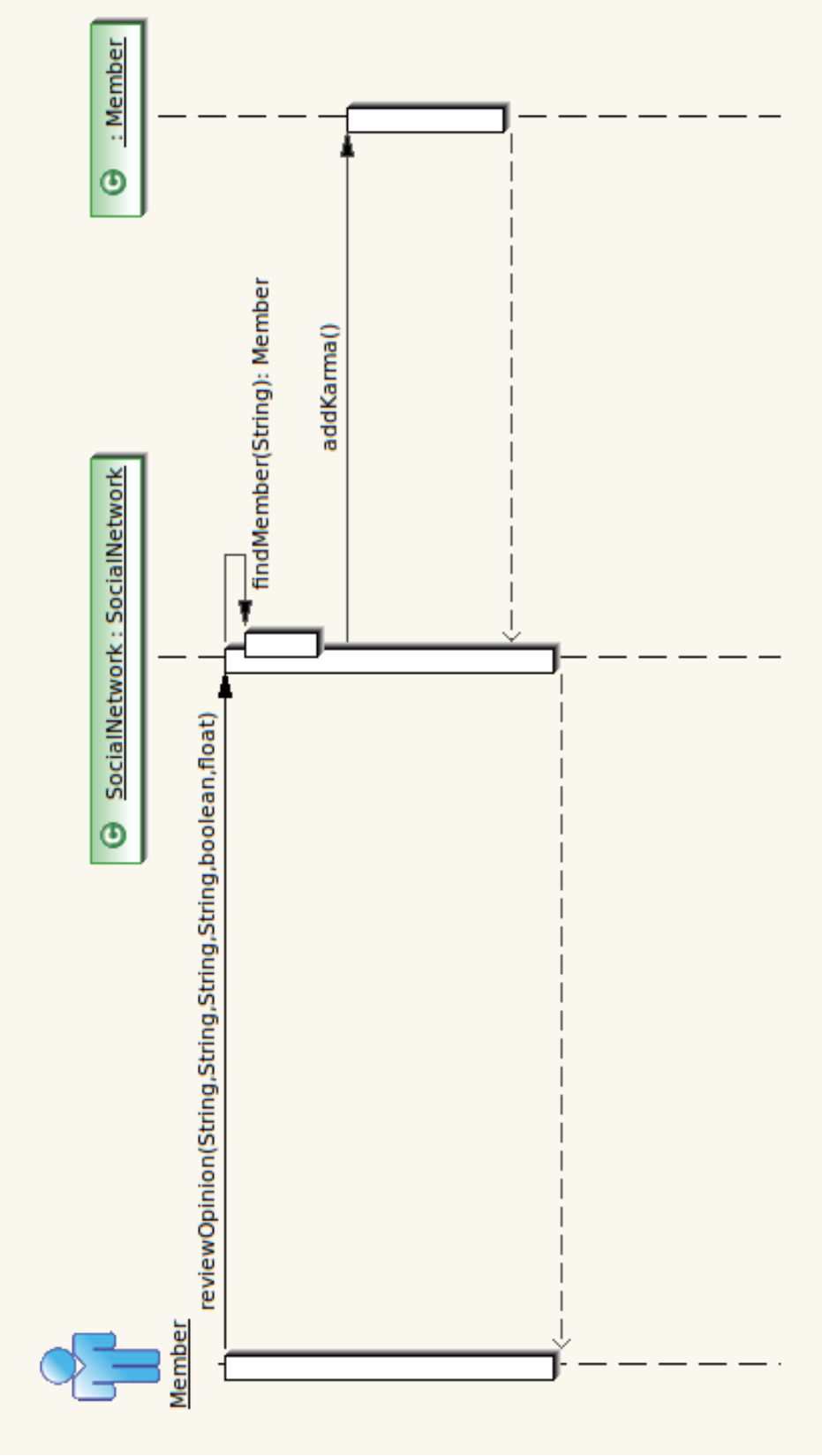

Figure 2 – Diagramme de séquence de reviewOpinion

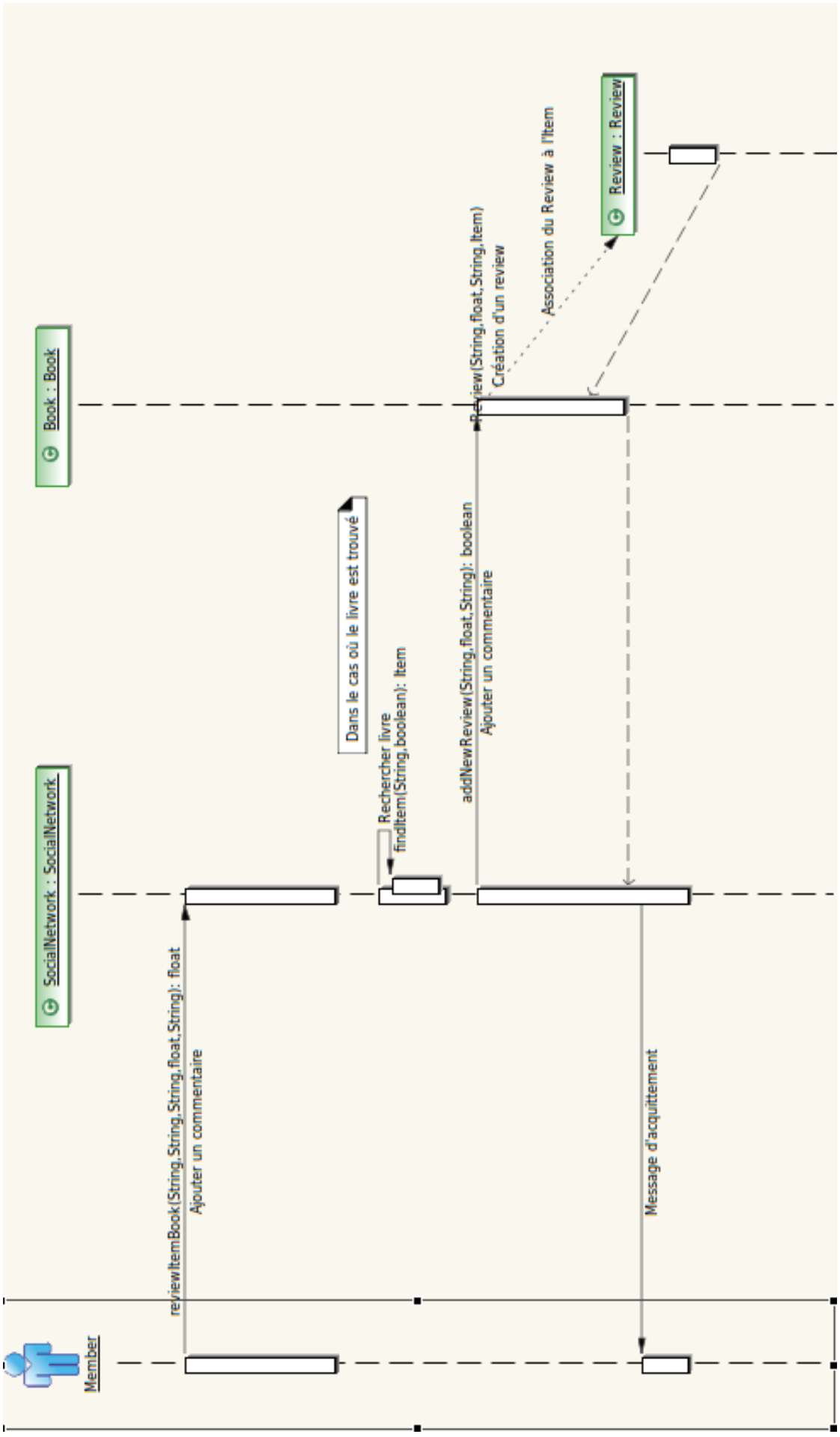

Figure 3 – Diagramme de séquence de reviewItemBook

Rapport d'audit

Le rapport d'audit effectué par un autre binôme nous a amené à changer notre code. Au total, se sont 8 remarques qui nous ont été faites. 7 d'entre elles concernaient un problème de fond et 1 un problème de forme. Nous avons acceptés 7 remarques. La remarque que nous avons refusée ne nous semblait pas opportune (gravité mineure).

La remarque de forme était très pertinente. En effet, le problème remonté était que nous utilisions toujours les mêmes conditions pour tester la validité des paramêtres donnés (*pseudo*, *password*, etc). Le copier-coller, rendant la tâche facile, nous à permis de réutiliser les condions des blocs *if* aisément. Cependant, il est plus ingénieux et réfléchis de créer des fonctions qui renvoient *true* ou *false* selon un paramêtre donné. Ainsi, s'il faut modifier la manière dont est testé le *password* par exemple, il suffit de modifier la fonction. Toutes les conditions utilisant cette fonction seront donc à jour, contrairement à la méthode que nous avions utilisée.

Les autres remarques, bien que pertinentes, n'étaient pas très graves et relevaient la plupart du temps d'un manque d'explications dans notre code.

Rapport de recette

Nombre de tests prévus	Tests passés	Tests OK	Tests Non OK	Tests NS	Anomalies majeures	Anomalies mineures
8	8	8	0	0	0	1

Numéro	Identification du test	Résultats	Gravité
1	Test en environnement «prestataire» : avec instructions du fichier «alire».	OK, pas d'erreur, mais il n'est pas possible de tester le lot2 depuis l'IHM	Mineure
2	TestsInitialisation	ok	RAS
3	TestsAddMember	ok	RAS
4	TestsAddItemFilm	ok	RAS
5	TestsAddItemBook	ok	RAS
6	TestsReviewItemFilm	ok	RAS
7	TestsReviewItemBook	ok	RAS
8	TestsReviewOpinion	ok	RAS

Lors de cette recette, aucune erreur importante n'a été relevée par le binôme client. Aucune modification n'a donc été apportée. Le fait que nos tests, ainsi que les tests du binôme client, ne relèvent aucune erreur nous permet donc de s'assurer de la conformité de notre livrable par rapport au cahier des charges.

Ci-dessous les fiches de tests du binôme correcteur.

Fiche de test N3

Objectif du test : éprouver la méthode addItemFilm sur les cas d'anomalie. Vérification de la levée de l'exception BadEntry lors de l'utilisation de AddItemFilm « avec des paramètres d'entrées incorrects ». **Pour rappel** : Les 5 cas de levée de l'exception BadEntry par la méthode addItemFilm prévus dans l'API sont :

- Si le pseudo n'est pas instancié ou a moins de 1 caractère autre que des espaces.
- Si le password n'est pas instancié ou a moins de 4 caractères autres que des leadings or traling blanks.
- Si le titre n'est pas instancié ou a moins de 1 caractère autre que des espaces.
- Si le genre n'est pas instancié.
- Si le réalisateur n'est pas instancié.
- Si le scénariste n'est pas instancié.
- Si la durée n'est pas positive

Description (scénario) :

- Instanciation d'un Social Network
- 1.1 Utilisation de AddItemFilm avec en paramètre un pseudo non instancié.
- 1.2 Utilisation de AddItemFilm avec en paramètre un pseudo ne contenant que des espaces.
- 1.3 Utilisation de AddItemFilm avec en paramètre un titre non instancié.
- 1.4 Utilisation de AddItemFilm avec en paramètre un titre ne contenant moins d'un caractère autres que des leadings or traling blanks.
- 1.5 Utilisation de AddItemFilm avec en paramètre un genre non instancié.
- 1.6 Utilisation de AddItemFilm avec en paramètre un réalisateur non instancié.
- 1.7 Utilisation de AddItemFilm avec en paramètre un scénariste non instancié.
- 1.8 Utilisation de AddItemFilm avec en paramètre une durée négative.

Résultats attendus : Dans les 8 cas d'utilisations de AddItemFilm, l'exception BadEntry doit être levée. De plus, après la levée de l'exception BadEntry lors de l'utilisation de AddItemFilm, le nombre de films doit rester identique au nombre de Film avant l'utilisation de AddItemFilm.

Résultats observés :

Les 8 cas d'utilisation lèvent bien une exception BadEntry comme prévu, le nombre de films n'a pas été modifié.

Conclusion :

Les tests sont concluants, l'application respecte le CdC.

Figure 4 – Fiche de tests numéro 3

Fiche de test N4

Objectif du test : éprouver la méthode AddItemFilm sur les cas de fonctionnement standard.

- But principal : Vérification de l'ajout d'un nouvel item de film après l'utilisation de AddItemFilm après un fonctionnement « avec des paramètres d'entrées corrects ».
- Buts secondaires :
  - Vérification de la levée d'exception NotMember par la méthode AddItemFilm dans tous les cas prévu dans l'API.
  - Vérification de la levée d'exception ItemFilmAlreadyExists par la méthode AddItemFilm dans tous les cas prévu dans l'API.

Description (scénario) :

- Instanciation d'un Social Network
- 2.1 Utilisation de AddItemFilm pour l'ajout de plusieurs Item « avec des paramètres d'entrées corrects »
- 2.2 Utilisation de AddItemFilm avec en paramètre un pseudo non existant.
- 2.3 Utilisation de AddItemFilm avec en paramètre un couple login/password non cohérent.
- 2.4 Utilisation de AddItemFilm avec en paramètre un item film de même titre déjà présent (mais avec des leadings et trailings blanks).

Résultats attendus :

L'ajout d'item de Film « avec des paramètres d'entrée corrects » ne doit pas lever d'exception, le nombre d'items de Film après ces ajouts doit avoir augmenté en conséquence. Dans les 2 utilisations suivantes de AddItemFilm, l'exception NotMember doit être levée. Dans la dernière utilisation, l'exception ItemFilmAlreadyExists doit être levée.

De plus, après la levée de ces deux exceptions, le nombre d'items de Film doit rester identique au nombre d'items de Film avant l'appel de la fonction AddItemFilm.

Résultats observés :

Les résultats sont en concordance avec ce qui était attendu.

Conclusion :

Le code respecte le CdC.

Figure 5 – Fiche de test numéro 4

Fiche de test N11

Objectif du test : éprouver la méthode reviewOpinion sur les cas d'anomalie.
Vérification de la levée de l'exception BadEntry lors de l'utilisation de reviewOpinion « avec des paramètres d'entrées incorrects ». Pour rappel : Les 5 cas de levée de l'exception BadEntry par la méthode reviewOpinion prévus dans l'API sont :

- Si le pseudo n'est pas instancié ou a moins de 1 caractère autre que des espaces.
- Si le password n'est pas instancié ou a moins de 4 caractères autres que des leadings or trailing blanks.
- Si le titre n'est pas instancié ou a moins de 1 caractère autre que des espaces.
- Si le genre n'est pas instancié.
- Si le réalisateur n'est pas instancié.
- Si le scénariste n'est pas instancié.
- Si la durée n'est pas positive.

Description (scénario) :

- Instanciation d'un Social Network
- 1.1 Utilisation de ReviewOpinion avec en paramètre un pseudo non instancié.
- 1.2 Utilisation de ReviewOpinion avec en paramètre un pseudo ne contenant que des espaces.
- 1.3 Utilisation de ReviewOpinion avec en paramètre un titre non instancié.
- 1.4 Utilisation de ReviewOpinion avec en paramètre un titre ne contenant moins d'un caractère autres que des leadings or traling blanks.
- 1.5 Utilisation de ReviewOpinion avec en paramètre un genre non instancié.
- 1.6 Utilisation de ReviewOpinion avec en paramètre un réalisateur non instancié.
- 1.7 Utilisation de ReviewOpinion avec en paramètre un scénariste non instancié.
- 1.8 Utilisation de ReviewOpinion avec en paramètre une durée négative.

Résultats attendus :
Dans les 8 cas d'utilisations de ReviewOpinion, l'exception BadEntry doit être levée. De plus, après la levée de l'exception BadEntry lors de l'utilisation de ReviewOpinion, le nombre de films doit rester identique au nombre de Film avant l'utilisation de ReviewOpinion.
Résultats observés :
Les exceptions sont levées correctement.
Conclusion :
Le CdC est respecté.

Figure 6 – Fiche de test numéro 11

Fiche de test N12

Objectif du test : éprouver la méthode reviewOpinion sur les cas de fonctionnement standard.
But principal : vérification de l'ajout d'une opinion après l'utilisation de reviewOpinion avec des paramètres d'entrée corrects.
But secondaire : vérification des levées des exception NotItem, NotMember.
Description (scénario) :

- Instanciation d'un Social Network
- Utilisation de addMember pour l'ajout de plusieurs membres avec paramètres d'entrées corrects.
- Utilisation de addItemBook pour l'ajout de deux Items Book.
- Utilisation de addItemFilm pour l'ajout d'un Item Film.
- Utilisation de ReviewItemBook pour création d'une opinion sur les deux items Book.
- Utilisation de ReviewItemFilm pour création d'une opinion sur l'Item Film.
- 1.1 Utilisation de ReviewOpinion avec paramètres d'entrée corrects pour review des trois items crées.
- 1.2 Utilisation de ReviewOpinion avec en paramètre un Membre inexistant.
- 1.3 Utilisation de ReviewOpinion avec en paramètre un titre d' Item inexistant.
- 1.4 Utilisation de ReviewOpinion avec en paramètre un pseudo de « reviewer » inexistant.

Résultats attendus :
Dans les 3 derniers cas d'utilisations de ReviewOpinion, les exceptions NotItem et NotMember doivent être levées. L'utilisation standard de ReviewOpinion doit modifier le karma de l'auteur de la review pour l'item concerné.
Résultats observés :
Les résultats observés sont conformes.
Conclusion :
Le code respecte le CdC.

Figure 7 – Fiche de test numéro 12

Fiche de test N5

Objectif du test : éprouver la méthode reviewItemFilm sur les cas d'anomalie. Vérification de la levée de l'exception BadEntry lors de l'utilisation de reviewItemFilm « avec des paramètres d'entrées incorrects ». **Pour rappel :** Les 5 cas de levée de l'exception BadEntry par la méthode reviewItemFilm prévus dans l'API sont :

- Si le pseudo n'est pas instancié ou a moins de 1 caractère autre que des espaces.
- Si le password n'est pas instancié ou a moins de 4 caractères autres que des leadings or traling blanks.
- Si le titre n'est pas instancié ou a moins de 1 caractère autre que des espaces.
- Si la note n'est pas comprise entre 0.0 et 5.0
- Si le commentaire n'est pas instancié.

Description (scénario) :

- Instanciation d'un Social Network
- Instanciation de trois utilisateurs
- Instanciation de trois films
- 1.1 Utilisation de reviewItemFilm avec en paramètre un pseudo non instancié.
- 1.2 Utilisation de reviewItemFilm avec en paramètre un pseudo ne contenant que des espaces.
- 1.3 Utilisation de reviewItemFilm avec en paramètre un password non instancié.
- 1.4 Utilisation de reviewItemFilm avec en paramètre un password contenant moins de 4 caractères autres que des leading et trailing blanks.
- 1.5 Utilisation de reviewItemFilm avec en paramètre un titre non instancié.
- 1.6 Utilisation de reviewItemFilm avec en paramètre un titre ne contenant que des espaces .
- 1.7 Utilisation de reviewItemFilm avec en paramètre une note négative
- 1.8 Utilisation de reviewItemFilm avec en paramètre une note supérieure à 5.0
- 1.9 Utilisation de reviewItemFilm avec en paramètre un commentaire non instancié.

Résultats attendus : Dans les 9 cas d'utilisations de reviewItemFilm, l'exception BadEntry doit être levée. De plus, après la levée de l'exception BadEntry lors de l'utilisation de reviewItemFilm, le nombre de films doit rester identique au nombre de Film avant l'utilisation de reviewItemFilm.

Résultats observés : Résultats conformes.

Conclusion : CdC respecté.

Figure 8 – Fiche de test numéro 5

Fiche de test N6

Objectif du test : éprouver la méthode reviewItemFilm sur les cas de fonctionnement standard.

- But principal : Vérification de l'ajout ou de la mise à jour d'une opinion concernant l'item film après l'utilisation de reviewItemFilm après un fonctionnement «avec des paramètres d'entrées corrects ».
- Buts secondaires : Vérification de la levée d'exception NotMember par la méthode reviewItemFilm dans tous les cas prévus dans l'API.
- — Vérification de la levée d'exception NotItem par la méthode reviewItemFilm dans tous les cas prévu dans l'API.

Description (scénario) :

- Instanciation d'un Social Network
- 2.1 Utilisation de reviewItemFilm pour l'ajout de plusieurs opinions « avec des paramètres d'entrées corrects »
- 2.2 Utilisation de reviewItemFilm avec en paramètre un pseudo non existant.
- 2.3 Utilisation de reviewItemFilm avec en paramètre un couple login/password non cohérent.
- 2.4 Utilisation de reviewItemFilm avec en paramètre un item film de titre inexistant.

Résultats attendus :
L'ajout d'une opinion sur un Film « avec des paramètres d'entrées corrects» ne doit pas lever d'exception. La nouvelle opinion saisie constitue la mise à jour de l'opinion. Dans les 2 utilisations suivantes de reviewItemFilm, l'exception NotMember doit être levée. Dans la dernière utilisation, l'exception NotItem doit être levée. De plus, après la levée de ces deux exceptions, l'opinion du membre concernant l'item film, existante ou pas, doit être identique à l'opinion avant l'appel de la fonction reviewItemFilm.
Résultats observés : Les résultats sont conformes. **Conclusion :** Le CdC est respecté.

Figure 9 – Fiche de test numéro 6

Bilan

Passons maintenant au bilan non fonctionnel du projet. Pour évaluer le rendement de notre programme, nous avons codé une classe *TestEfficient* qui permet de mesurer le temps d'exécution des méthodes clés du projet (i.e celles utilisées directement par l'IHM). Ces méthodes sont appelées au sein d'un *SocialNetwork* contenant 500 membres et 5000 items. Ci-dessous un exemple de code écrit :

Listing 1– Code de rendement d'une fonction

```
...
double duree;
//Timer
double start;
start = System.nanoTime();
try{
        sn.addMember("toto", "password", "nouveau membre");
}
catch(BadEntry e)
{
        System.out.println("BadEntry");
}
catch(MemberAlreadyExists e)
{
System.out.println("MemberAlreadyExists");
}
duree = System.nanoTime() - start;
System.out.println("Duree pour ajouter un nouveau membre : " + duree/1000000000 + " secondes\n");
...
```

Les résultats obtenus suivants sont tout-à-fait concluant et respectent le cahier des charges.

Listing 2– Resultats obtenus

```
Duree pour ajouter un nouveau membre : 8.0896E-5 secondes
Duree pour ajouter un film : 3.88864E-4 secondes
Duree pour ajouter un livre : 0.001121024 secondes
Duree pour rechercher un item : 0.001814016 secondes
Duree pour ajouter un avis sur un Film : 0.00116992 secondes
Duree pour ajouter un avis sur un Book : 0.001432064 secondes
Duree pour ajouter un opinion(karma) a un membre : 0.001236992 secondes
```

Bilan général

Pour dresser un bilan général du travail réalisé, nous pouvons tout d'abord dire que nous avons réussi à s'organiser au sein du binôme pour se répartir au mieu les tâches. De plus, l'utilisation de solution de «versioning» à grandement facilité le partage et la mise en commun du code.

- *Quels sont les points faibles et les points forts de votre livrable ?*

Le livrable que nous proposons est, d'une part, simple à comprendre grâce à la javadoc et, d'autre part, fidèle au cahier des charges grâce notamment à la batterie de tests effectués sur celui-ci. En plus des tests «classiques» sur les méthodes et les classes qui vont être utilisées, nous avons effectués des tests de rendement sur l'utisalisation de celle-ci.

- *Si c'était à refaire, que changerions-nous, le cas échéant, dans notre démarche de conception-réalisation ?*

Pour reprendre la remarque relevée lors du rapport d'audit, nous pouvons dire qu'une des erreurs que nous avons faite a été de commencer le codage des méthodes trop rapidement. En soit, le premier code proposé n'était pas mauvais, mais il n'était pas correctement structuré. En effet, un temps de réflexion au préalable, sur les fonctions et les conditions qui allaient être appelées fréquement, nous aurait permis de gagner du temps en fin de projet.

- *Quels sont les avantages et inconvénients trouvés dans la démarche de conception en W ?*

La conception en W possède quelques avantages, comme par exemple le fait, qu'en suivant ce cycle, nous sommes obligés de passer du temps sur l'analyse et sur la conception. Le temps passé lors de ces deux phases nous a permit de s'assurer que la phase de réalisation n'allait pas être vaine, et que ce que nous allions faire correspondrait bien à la demande. De plus, les phases de tests (unitaires et d'intégration) permettent de valider au fur et à mesure les différentes parties du projet.

Un inconvénient ...

Annexe A

Fiches de tests