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 **3**

Fiches tâches/temps **4**

Diagramme de classes et diagrammes de séquences **6**

Rapport d'audit **10**

Rapport de recette **11**

Bilan **12**

Bilan général **13**

Annexes **14**

Introduction

Ce rapport a pour but de dresser un bilan le plus complet possible du projet mené en INF112. Vous trouverez dans ce rapport les différentes informations concernants 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.

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 87 heures de travail, nous chiffrons le coût de notre activité à 5220 €.

- *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).

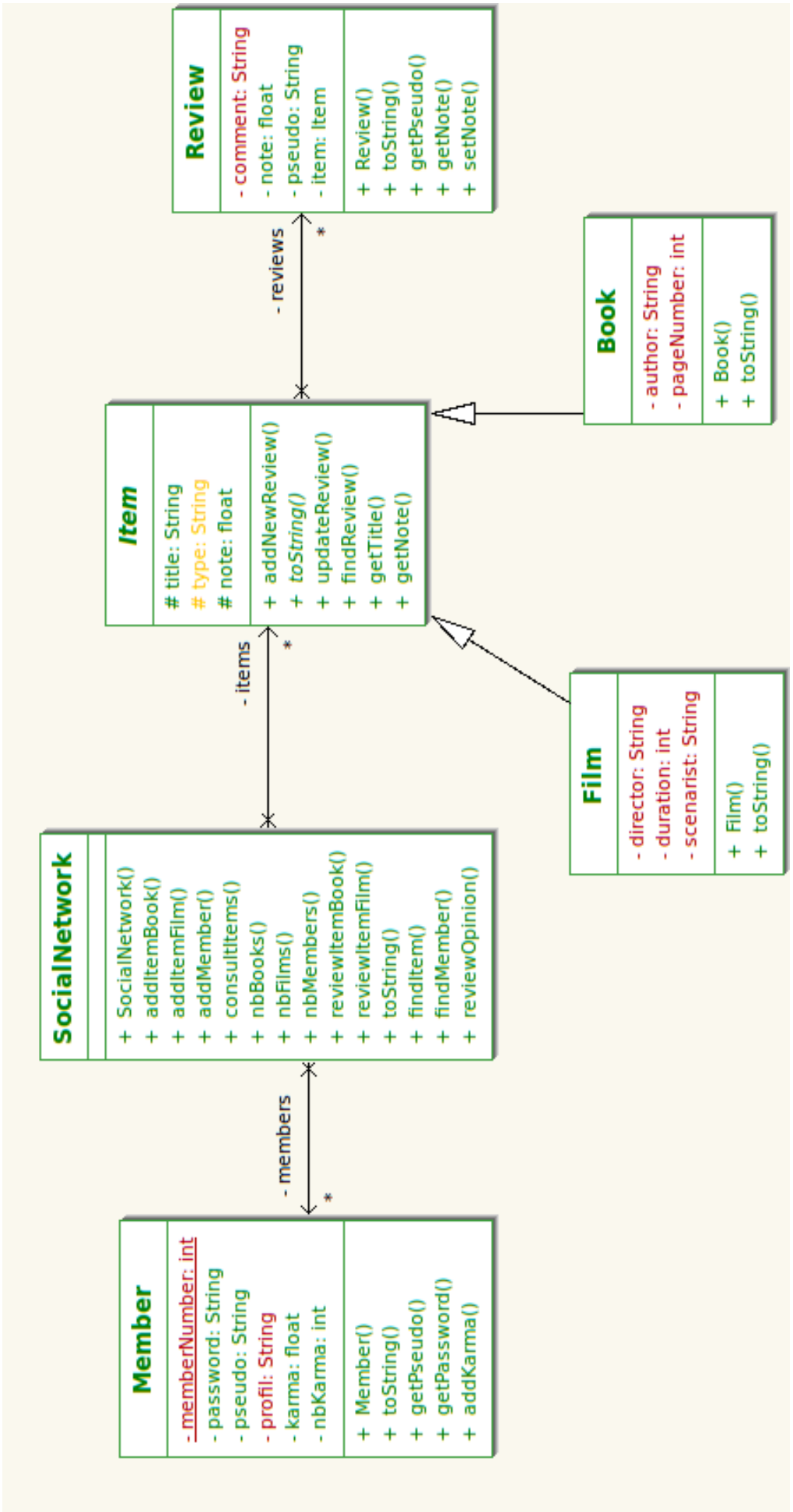

Figure 1 – Diagramme général de classes

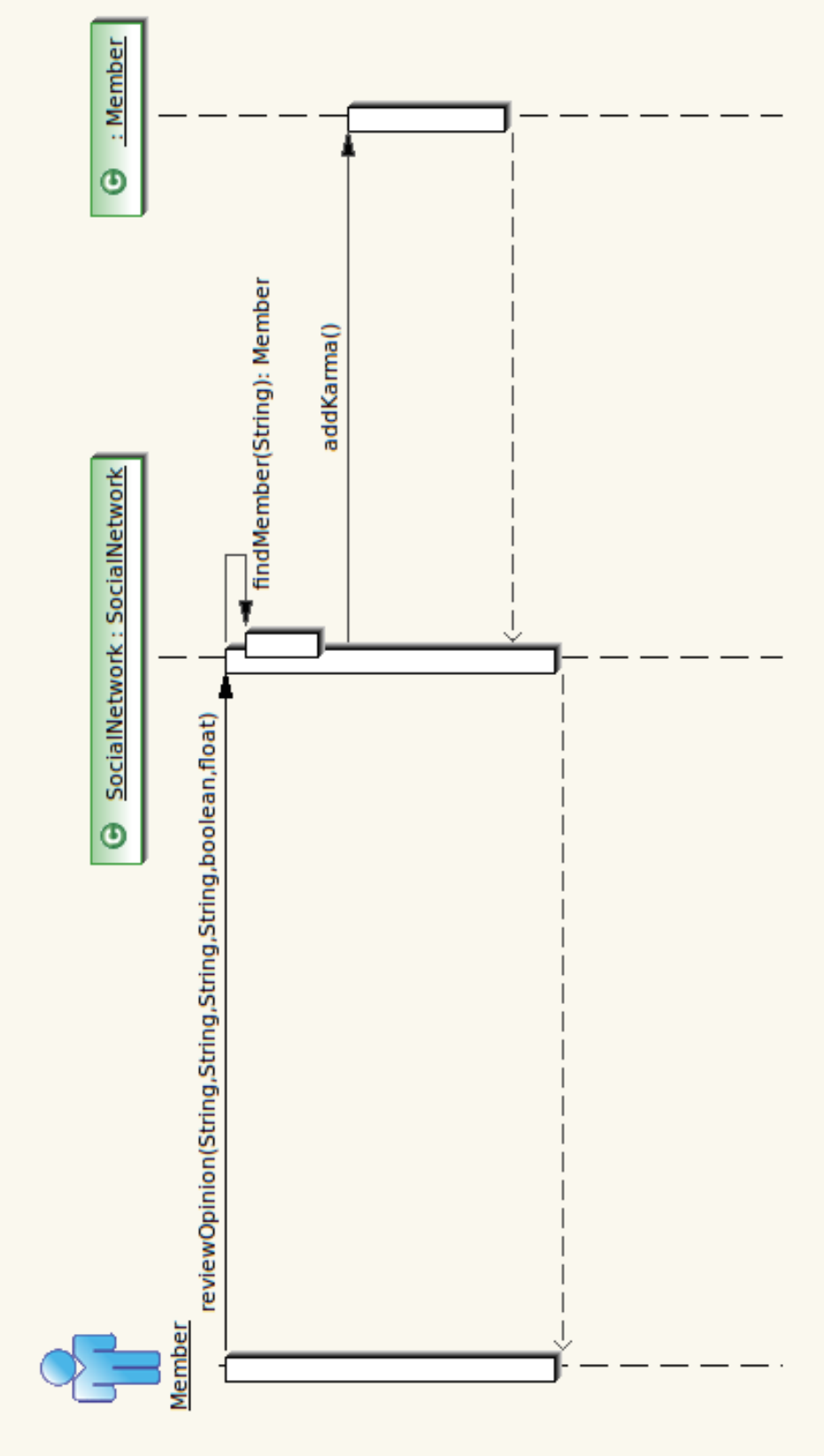

Figure 2 – Diagramme de séquence de reviewOpinion

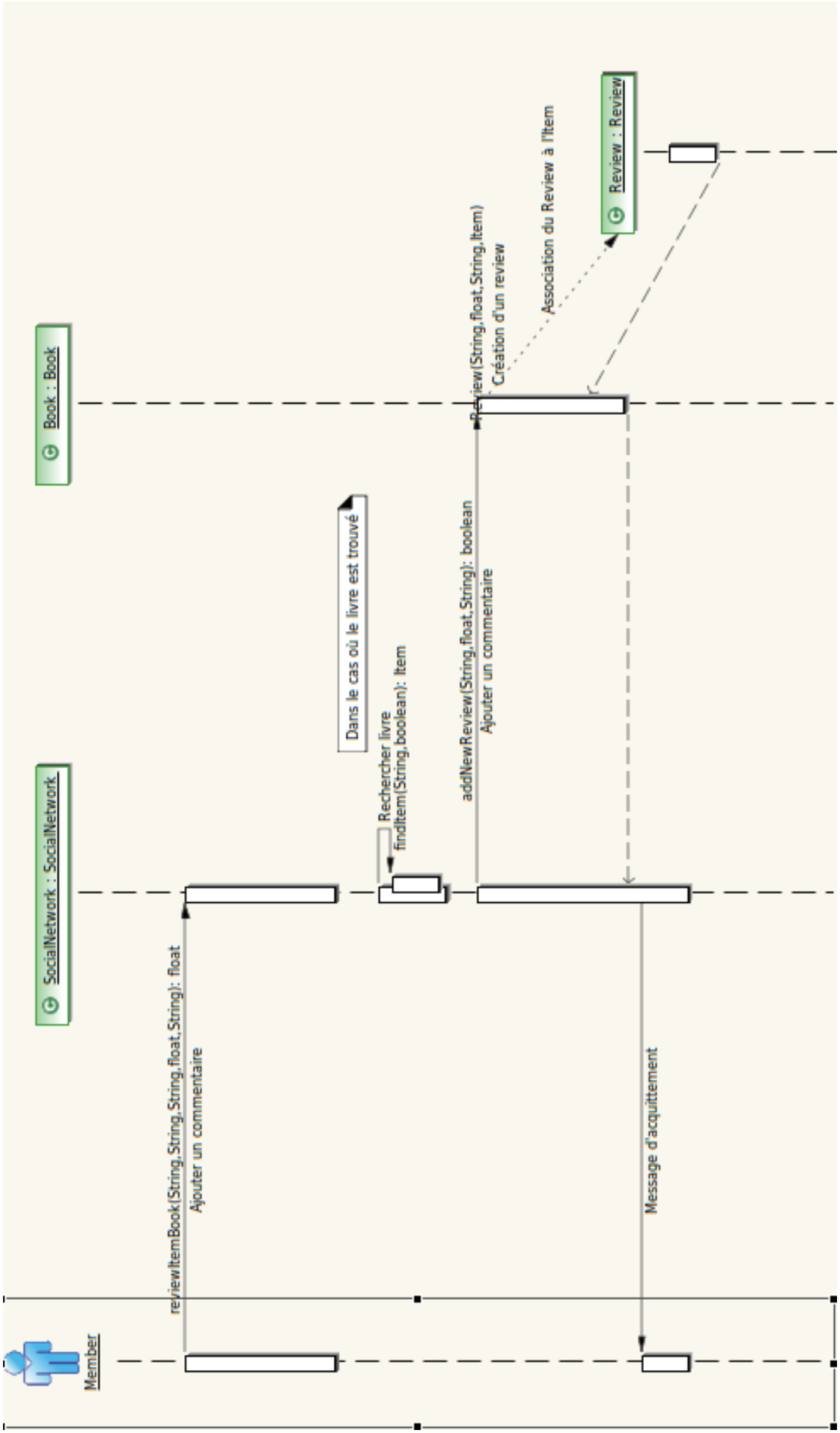

Figure 3 – Diagramme de séquence de reviewItemBook

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

Lors de cette recette, aucune erreur 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.

Bilan

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 ?*

Annexes