université de BORDEAUX

PdP - Métriques de Maintenabilité

Compte rendu de TD

Jeudi 4 Avril 2019

Dépôt Savane :
`https://services.emi.u-bordeaux.fr/projet/savane/projects/pdp2019mm/`

Soumis pour :
(Client) Narbel Philippe
(Chargé de TD) Hofer Ludovic

Soumis par :
Delrée Sylvain
Giachino Nicolas
Martinez Eudes
Ousseny Irfaane

1 TD - 23 janvier

Réunion autour de la mise en place du cahier des besoins

— Utilisation de verbe d'action afin de décrire les besoins fonctionnels
— Spécification de la dépendance (au sein des classes, packages, attributs, méthodes,...)
— Extraction des données à partir d'une des méthodes (analyse de bytecode ou de code source), choix d'implémentation à justifier.
— Tests : garantit la fiabilité de l'implémentation
— Détailler les termes "Bons codes / Mauvais code", si possible les illustrer avec des exemples.
— Documentation (à demander au client) et format du résultat souhaité (graphe, résultat affiché dans un fichier, navigateur,...)
— Test de charge, pour le temps d'exécution du problème, est-on limiter à des contraintes de temps ? A voir avec le client.
— Classifier les besoins par priorités (À l'aide de macros LaTeX)

2 TD - 30 janvier

Discussion à propos de la rédaction du cahier des besoins
De nombreux soucis ont été mis en lumières :

— Eviter d'employer des termes avant de les décrire, il faut décrire les concepts avant de les évoquer
— Description de la métrique de Martin : il faut la remettre au centre du projet
— Revoir la distinction classes / catégories
— On parle du langage JAVA après la description de JDepend

Dans le cadre des calculs de dépendances, il faut bien preciser que la métrique de Martin n'est pas un critère assurant la maintenabilité d'un logiciel (ce n'est pas une valeur absolue) mais un indicateur qui entre dans le calcul.

Des oublis ont été faits dans la description des besoins :

— L'extraction des dépendances qui est le point fondamental du projet n'a pas été abordé
— Comment allons-nous extraire les dependances ?
— Differents types de dépendances : extraction de dépendances classe à classe. Extraire la liste des méthodes et la liste des méthodes abstraites pour le calcul de l'abstraction

Bilan du TD2 : revoir l'agencement / ordonnencement de certaines parties afin d'assurer une cohérence dans la lecture et du domaine au niveau de l'introduction

3 TD - 5 Février

Retour sur le cahier des besoins après le rendu

Points négatifs

- Contradictions dans les définitions, distinction entre la terminologie commune n'est pas la même que la terminologie employée dans le cadre de la métrique de Martin.
- Définitions multiples de concepts
- Problème au niveau du formalisme qui n'est pas assez détaillé, expliqué, de plus la formule du couplage efferent (Ce) est fausse (ou ne prend pas en compte la notion de catégorie)
- Pas de précision sur l'analyse syntaxique, les besoins extrait ne sont pas clair
- Manque d'analyse sur ce qui est faisable ou pas
- Manque de numérotations des besoins

Points positifs

- Bonne compréhension du sujet, lecture agréable
- Bien d'avoir pensé à faire un formalisme pour le calcul des métriques malgré quelques coquilles
- Explication claire de JDepend

4 TD - 13 Février

Discution autour du code avant le rendu de la première release.

Malgré un code bien organisé, quelques remarques ont été faites :

- Manque d'un README afin de comprendre les modalités de compilation et d'exécution.
- Choix d'analyse des dépendances se base sur le bytecode JAVA, pourquoi ne pas avoir commencer l'analyse syntaxique au niveau du code source JAVA en prevision de la première release.
- Manque de tests unitaire sur les fonctions
- Gestion des warnings n'a pas été réalisé, 130 warnings ont été relevés, nous utilision IntelliJ comme IDE et il nous masque les warnings) d'où la non-gestion de ceux-ci.
- Concernant le graphe des dépendances, la police devrait être plus grande, l'espace entre les cadres devraient être réduits

5 TD - 6 Mars

Retour sur l'audit :

- Problème au niveau de la gestion du temps
- Trop absolue sur la métrique!! Ne pas oublier que c'est un indicateur
- Certaines diapositives étaient "vides"
- L'exemple projeté était pas lisible, il aurait fallu, au préalable, présenter un exemple avec peu de sommets

Retour sur le code de la première release :
— Gestions d'erreurs lié aux commandes décrit dans le fichier README
— Dans le fichier gradle, la *mainClass* n'a pas été renseigné (JAR)
— On peut négliger l'extension *".class"* dans le nom des sommets des graphes
— L'architecture proposé est bonne

Discussion autour des améliorations possibles :
— Analyse de code source JAVA
— Regrouper les métriques au niveau d'une classe unifiée
— Correction du cahier des besoins

6 TD - 13 Mars

Discussion autour de la sortie des résultats et des problèmes d'exécution :
— Ecrire un script sur Python (ou R) afin d'afficher les résultats sous la forme d'un histogramme
— Retravailler le graphe des dépendances afin d'afficher les dépendances externes (Non voulu par le client)
— Echec de deux tests, lié aux path que l'on récupère au niveau de la méthode *generateStructure*
— Revoir les options des lignes de commandes pour l'exécution au niveau du fichier README et fournir une documentation lors du lancement de l'application expliquant les differentes possibilités de traitements

7 TD - 27 Mars

Réunion autour du code et du mémoire avant le rendu final.
A propos du code :
— Spécifier les arguments au niveau de certains scripts Python
— Ajouter un fichier *Makefile* et d'un *.gitignore* pour le code LaTeX
— Flexibilité du package *metrics* afin de pouvoir par la suite ajouter de nouvelles métriques
A propos du mémoire :
— Ajouter un recapitulatif des besoins, mettre en évidence ceux qui ont été réalisés et ceux qui n'ont pas été résolues.
— Apporter des critiques vis-à-vis des graphes / histogrammes
— Renommer la partie *Terminologie* en *Concepts* aurais plus de sens
— Positionnement du terme de granularité dans le mémoire
— Garder le concept de catégoie définit dans le mémoire en expliquant qu'elle est renseigné dans l'article de base de Martin (publié en 1994), puis dans son livre parut en 2003, il ne mentionne plus le terme de catégorie mais du terme de package.
— Décrire les tests de rupture peut être une bonne chose à ajouter dans le partie "Ce qui ne passe pas" du mémoire