Projet Design Pattern

Master 1 Informatique - 2005-2006
Rémi Forax, Gautier Loyauté & Dominique Revuz
forax@univ-mlv.fr, loyaute@univ-mlv.fr, dr@univ-mlv.fr

Le projet

Ce projet est décomposé en deux parties distinctes ayant chacune une date de rendu différente.
La première partie consiste:

- à corriger un bug existant dans le JDK du langage Java.
- à fournir sous forme d'un diagramme de Gantt un plan de travail prévisonnel de la seconde partie du projet.

La seconde partie consiste:

- à effectuer un travail d'analyse *à postériori* d'un logiciel existant
- à écriture le code permettant d'effectuer un certain nombre de modifications mineures sur ce logiciel.

Corriger un bug du JDK

L'idée est de vous apprendre à corriger un bug dans un code existant à partir d'un rapport de bug. Nous utiliserons pour cela le JDK de SUN qui contient quelques bug, si,si.

Trouvez un bug à corriger en utilisant l'interface en ligne situé à cette adresse :
http://bugs.sun.com/bugdatabase/index.jsp

Pour éviter qu'un bug ne soit corrigé plusieurs fois et pour vous aider à ne pas prendre des bug trop facile ou trop difficile, il est demandé d'envoyer le numéro de bug que vous souhaité corriger par mail à Rémi Forax (forax@univ-mlv.fr).
Vous pourrez commencer à travailler dessus qu'après avoir reçu un mail de confirmation.
Pour aider vos petits camarades avant d'envoyer votre demande de validation vérifier que le bug n'a pas été choisi en vérifiant dans le forum
http://ankh.univ-mlv.fr/~metaone/cplusplus/index.php
et postez votre numéro de bug une fois celui-ci choisi.

Pour effectuez la modification/correction technique du bug, récupérer la dernière version des sources du JDK https://mustang.dev.java.net/ et à effectuer les changements dans la ou les classes.
Attention à garder une compatibilité ascendante avec le code existant.
Tous les bugs consistant à ajouter une nouvelle classe seront considéré comme trop complexe.

Enfin créer la correction de bug qui correspond au documents suivants :

- la fiche du bug original ;
- une analyse en francais de la correction proposée ;
- le code des classes corrigés ;
- un **diff -u** entre les classes originales et les classes corrigés (patch)
- un test unitaire utilisant **JUnit** montrant la correction effective du bug et la non-destruction de la compatibilité avec le code existant.

Il est vous demandé de corriger un bug par étudiant. Vous pouvez si vous le souhaitez vous pouvez par contre travail à deux dessus, soit deux bugs par binome.

Planification de la Deuxième partie
Vous devez fournir une planification de la deuxième partie (Attention ceci demande au moins une bonne heure de travail très énergique).
Première étape, découper le travail à faire en tâches, une tâche prend au maximum 2 heures et est définie par l'objectif à atteintre à la fin de la tâche.
Une fois ce travail fait (découpage en tâches), faire de diagramme d'activité Gantt des deux binôme sur la période du 2 janvier au 6 fevrier indiquant les jours ou vous avez prévu de travailler pour chaque binome et sur quel tâche.
Une tâche peut être assigner aux deux binomes ou à un seul suivant sa difficultée.

Analyse de code

Le logiciel Tatoo (disponible à cet adresse http://monge.univ-mlv.fr/~roussel/Tatoo/) est un compilateur de compilateur écrit en Java par Julien Cervelle, Rémi Forax et Gilles Roussel.
Le but de cette partie consiste à analyser le logiciel pour effectuer un ensemble de modifications.
Il vous est donc demandé d'effectuer une documentation développeur devant inclure

- une introduction expliquant les différentes parties du logiciel et leur fonction ;
- d'indiquer pour chaque parties du logiciel les paquetages Java correspondant
- faire un diagramme de classe pour chaque paquetage et commenter en francais chaque diagramme en indiquant :
    - la fonction de chaque classe
    - les design-patterns présents en indiquant pourquoi ce design-pattern a été utilisé (vous devriez trouver l'ensemble des design-patterns).
- de créer les diagrammes de séquence pour :
    - la gestion de la ligne de commande
    - la génération du lexer
    - la génération du parseur
    - la génération des tools
      ainsi que les diagramme de séquence pour :
    - l'analyse d'un texte en flot de token par le lexer
    - l'analyse d'une grammaire à partir des terminaux pour le parseur

  chaque diagramme de séquence devra être expliqué en français (au moins une page pour l'explication).

- Critiquer le code et d'indiquer les parties de codes qui pourrait poser problème ou qui contiennent des bugs ainsi que les codes identiques qui pourraient être partagés (5 pages au moins)
- de créer des tests unitaires (JUnit) pour chaque classe correspondant à une implantation. Chaque test devra tester **toutes** les méthodes de l'implantation.

Il vous est demandé de plus d'implanter les changements demandés suivants :

- remplacer les **XXX** par quelque chose d'approprié
- d'écrire une tâche **ant** permettant d'exécuter successivement le lexer, le parser et tools en une fois
  (l'équivalent **ant** de **fr.umlv.tatoo.cc.main.main.MainMain**).
- faites en sorte que le contexte de génération manipule les imports en utilisant la notion de **Type** (voir **fr.umlv.tatoo.cc.tools.generator.Type**) et pas de **String**. Il faudra sûrement changer la classe **Type** de paquetage.

Pour chaque changement, le document devra expliqué quel sont le (ou les) stratégies d'implementation proposée, celle que vous avez retenue et le code de celle-ci.

Enfin, il vous est demandé de produire un document de planification ayant le même format que celui transmis pour le premier rendu mais indiquant le temps réels passé pour réalisé chaque tâche.
Expliqué dans votre document les déviations qu'il a surement eut entre le planning prévisionnel et le planning réalisé, pourquoi.

Le tout devra consituer un document en français, claire et lisible du début à la fin et pas une compilation de documents.

Détail des rendus

Le premier et le second rendus devront impérativement être effectué par les mêmes binômes sous peine de zéro à l'ensemble des binômes.

Les documents du premier et second rendus devront être transmis au format PDF dans une archive ZIP par mail à l'ensemble des personnes indiqués en début de sujet avec dans le sujet de votre mail "[Master2] Design-pattern Tatoo binome %s %s"

Date limite du premier rendu : 23h59m59s 31 Décembre 2005 (bonne année :)
Date limite du second rendu : 23h59m59s 5 fevrier 2006.