INF5153 – Génie logiciel : conception

Refactoring et smells

Jacques Berger

Objectifs

Introduire la notion de refactoring

Discuter de smells

Prérequis

Patrons de conception

Refactoring

Améliorer la lisibilité du code source et la conception d'un logiciel sans changer sa fonctionnalité

Le refactoring est intimement lié à la conception

Quand?

«In our experience no set of metrics rivals informed human intuition.»

-Martin Fowler, Refactoring – Improving the Design of Existing Code

Smells

L'analogie de la «mauvaise odeur» est utilisée pour indiquer lorsque le code nécessite un refactoring

Le smell est un indicateur montrant une amélioration possible

Duplicated Code

Lorsqu'une même structure de code est utilisée à plusieurs endroits

Le code peut être centralisé à un seul endroit

Long Method

Plus une méthode est longue, plus elle est difficile à comprendre

La méthode peut être divisée en plusieurs petites méthodes

Long Method

Plusieurs petites méthodes peuvent amener une certaine complexité, facilement gérable avec une bonne nomenclature

Un nom de méthode peut servir à documenter un comportement plutôt que d'utiliser un commentaire

Large Class

Si une classe possède beaucoup de variables d'instance, il y a probablement de la duplication

Apparition de préfixes ou suffixes dans les noms des variables

Extraire une nouvelle classe

Long Parameter List

Si la liste de paramètres est trop longue ou change régulièrement, il faut repenser à la structure de dépendances

Divergent Change

Lorsqu'une classe change de différentes façons pour différentes raisons

Extraire une classe pour chaque raison de modifier la classe

Shotgun Surgery

Lorsqu'un changement entraîne la modification de plusieurs classes différentes

Probablement une duplication

Feature Envy

Lorsqu'une méthode s'intéresse trop au fonctionnement d'une autre classe

Par exemple, la méthode appelle plusieurs getters d'un autre objet pour effectuer un calcul

Déplacer le traitement au bon endroit

Data Clumps

Lorsqu'on observe toujours les mêmes données regroupées ensemble

Devrait devenir une classe

Primitive Obsession

Une donnée représentant un petit concept peut facilement devenir une classe

En OOP, il est plus facile de manipuler des objets que des types primitifs

Switch Statements

Le switch est un problème s'il est dupliqué

Le polymorphisme peut être une solution à ce problème

Cas par cas

Parallel Inheritance Hierarchies

Lorsqu'on crée une sous-classe et qu'on doit obligatoirement créer une autre sous-classe en parallèle

Lazy Class

Une classe qui ne fait rien devrait être éliminée

Speculative Generality

Éviter la mise en place de complexité non nécessaire

- Classes abstraites sans traitement
- Délégation non nécessaire
- Paramètres non utilisés
- Méthodes avec un nom étrange et abstrait

Temporary Field

Lorsqu'une variable d'instance n'est initialisée que dans certains scénarios précis

Comprendre l'utilité d'une telle variable est difficile

Considérer un remplacement pour une nouvelle classe

Message Chains

Une délégation qui fait une délégation qui fait une délégation...

Middle Man

Lorsque la moitié ou plus des méthodes d'une classe sont une délégation vers une autre classe

Parler directement à la classe concernée

Inappropriate Intimacy

Lorsqu'une classe utilise les membres privés d'une autre classe

L'héritage peut entraîner ce type de relation

Briser la dépendance entre les classes

Alternative Classes with Different Interfaces

Fusionner progressivement ces classes

Incomplete Library Class

Lorsqu'une classe d'une librairie n'offre pas une fonctionnalité qui vous serait essentielle

Souvent impossible à modifier directement (et modifier une librairie n'est pas une si bonne idée...)

Ajouter la fonctionnalité dans une classe qui utilise la librairie à l'interne

Data Class

Lorsqu'une classe ne contient que des données avec getters et setters mais aucune logique

Prendre une responsabilité

Refused Bequest

Lorsqu'une sous-classe ne veut pas utiliser certaines méthodes de la classe de base

Souvent un mauvais lien d'héritage

Plus grave si on refuse de supporter certaines interfaces de la classe de base

Comments

Souvent, les commentaires indiquent la présence de mauvais code

Lorsqu'on sent le besoin de commenter, d'abord tenter de régler le problème par le refactoring

Plus loin...

Refactoring: Improving the Design of Existing Code
Martin Fowler
Addison-Wesley, 1999