Fiche méthodologique de projet

C.6 Concevoir des objets en respectant les principes OO

Objectif

Appliquer le Faible couplage et la Forte cohésion

Pattern Faible Couplage

Le couplage est une mesure du degré auquel un élément est lié à un autre, en a connaissance ou en dépend. S'il y a couplage ou dépendance, l'objet dépendant peut être affecté par les modifications de celui dont il dépend. Un objet A qui fait appel aux opérations d'un objet B a un couplage aux service de B. Le faible couplage a pour objectif de faciliter la maintenance en minimisant les dépendances entre éléments.

Bien entendu, il ne faut pas tomber dans le piège de décider de concevoir des éléments tous indépendants et faiblement couplés, car ceci irai à l'encontre du principe OO définissant un système objet comme un ensemble d'objets connectés les uns aux autres et communiquant entre eux.

En général ce pattern est appliqué inconsciemment, c'est davantage un principe d'appréciation qu'une règle.

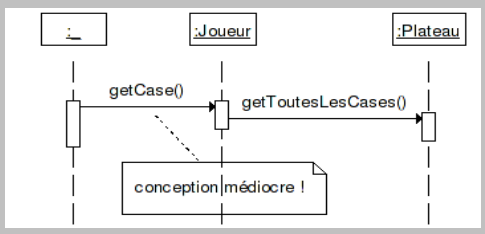
Nom	**Faible Couplage**
Problème	Comment réduire l'impact des modifications ?
Solution	Affecter les responsabilités de sorte à éviter tout couplage inutile

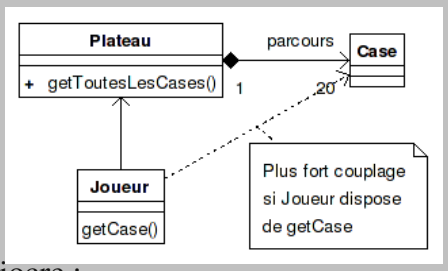
Exemple

Problème : pourquoi un Plateau plutôt qu'un Chien ?

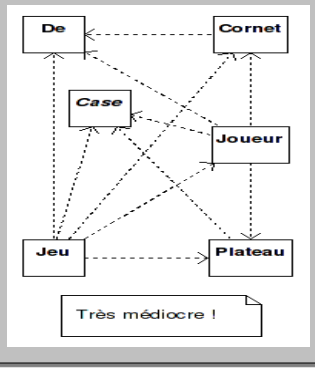
On a décidé d'affecter la responsabilité de connaître une Case particulière à l'objet Plateau parce que le Plateau sait tout sur les Cases (c'est lui l'Expert). Mais pourquoi Expert donne-t-il ce conseil ? Car dans ce cas, seul le Plateau sera couplé à Case.

Une autre (mauvaise) solution augmenterait le couplage :





Et alors on pourrait arriver à une solution très médiocre :



Pattern Forte Cohésion

La cohésion mesure le degré de spécialisation des responsabilités d'un composant ou classe. Comme dans le pattern Faible couplage, la cohésion médiocre altère la compréhension, la réutilisation, la maintenabilité et subit toute sorte de changements.

C'est souvent l'erreur commise par les développeurs OO débutants : confier tout le travail à une seule classe et on obtient un objet trop « gonflé » !

Par exemple on ne peut exiger d'un réfrigérateur qu'il fasse radiateur et range-disques en même temps (en tout cas dans sa modélisation logicielle). La multiplication des disciplines à responsabilité accroît exponentiellement le risque d'erreurs intrinsèques à cette classe.

Nom	**Forte Cohésion**
Problème	Comment s'assurer que les objets restent compréhensibles et faciles à gérer (et qu'il contribue au Faible Couplage) ?
Solution	Affecter les responsabilités pour que la cohésion demeure élevée.

Exemple

Problème : l'opération système lancerPartie est appelée, et après ?