Fiche méthodologique de projet

C.3

Mettre en oeuvre des patterns en conception préliminaire

Objectif

Utiliser des patterns GRASP pour la conception préliminaire.

Préambule

Une fois l'analyse terminée, on sait donc « quoi faire ». Maintenant en conception, on doit déterminer « comment le faire ». L'étape de conception peut se décomposer en deux parties : la conception préliminaire et la conception détaillée.

Les patterns

Dans le monde de l'orienté-objet, les design patterns se présentent comme un catalogue de méthodes de résolution de problèmes récurrents.

Un pattern ou motif de conception est un document qui décrit une solution générale à un problème qui revient souvent.

Les patterns GRASP

Les patterns GRASP sont des patrons créés par Craig Larman qui décrivent des règles pour affecter les responsabilités aux classes d'un programme orienté objet pendant la conception, en liaison avec la méthode de conception BCE (Boundary Control Entity), en français MVC (Modèle Vue Contrôleur).

Il y a 9 patterns GRASP : Créateur, Expert, Faible couplage, Contrôleur, Forte cohésion, Fabrication pure, Polymorphisme, Protection des variations, Indirection

Remarque : ces patterns ne sont qu'une aide pédagogique qui permet de structurer et de nommer des principes. Une fois que ces principes sont saisis, les termes spécifiques sont sans importance.

Méthode

Quelques exemples de questions que l'on se pose en conception :
A quelle classe faut-il confier la responsabilité de certains attributs ?
Qui est responsable du contrôle ... ? Qui coordonne l'ensemble ?
Doit-on créer une classe X ?
Qui instancie les objets x1 et x2 ? Qui lance l'objet x1 ?
etc ...

Pattern Créateur

C'est l'un des premiers problèmes à prendre en considération en conception Orientée Objet. C'est une **responsabilité de faire quelque chose**.

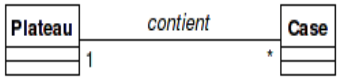
En fait n'importe quel objet pourrait créer un objet Dé ou un objet Case, mais lequel serait choisi par le plus grand nombre de développeurs OO expérimentés ? Et pourquoi ?
Pourquoi ne pas demander à un objet Chien d'être le créateur ! Non parce que cela ne cadre pas avec le modèle du domaine. De la même manière, on ne nommera pas les classes Joueur et Dé : AB123 et R2D2 !

Le pattern Créateur se définit de la manière décrite ci-dessous.

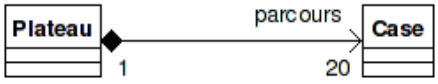
Nom	**Créateur**
Problème	Qui crée A ?
Solution	Affecte à la classe B la responsabilité de créer une instance de la classe A si une ou plusieurs des conditions suivantes est vraie (plus il y en a mieux c'est) : • contient ou agrège des objets A • enregistre des objets A • utilise étroitement des objets A • possède les données d'initialisation des objets A

Exemple

Problème : qui crée l'objet Case ?
A partir du modèle du domaine, on constate qu'un Plateau contient des Cases.



En cohérence avec le principe du pattern Créateur, le Plateau créera les Cases. De plus, les Cases feront toujours partie d'un Plateau et celui-ci gérera leur création et leur destruction : les Cases sont donc dans une relation de composition avec le Plateau.



Pattern Expert

L'objectif de ce pattern est d'**affecter au mieux une responsabilité à une classe logicielle**. Afin de réaliser ces responsabilités, cette classe doit disposer des informations nécessaires.

Si les responsabilités sont mal réparties, les classes logicielles vont être difficilement maintenables, plus dure à comprendre, et avec une réutilisation des composants peu flexible.

Ainsi, comme d'autres pattern GRASP, le modèle Expert est un pattern relativement intuitif, qui en général, est respecté par le bon sens du concepteur. Il s'agit tout simplement d'affecter à une classe les responsabilités correspondants aux informations dont elle dispose intrinsèquement (qu'elle possède) ou non (objets de collaborations).

Nom	**Expert en Information** ou plus simplement **Expert**
Problème	Quel est le principe général d'affectation des responsabilités aux objets ?
Solution	Affecter la responsabilité à la classe qui possède les informations nécessaire pour le faire

Exemple

Problème : qui connaît un objet Case à partir d'une clé ?
Pour s'acquitter de cette responsabilité, il faut pouvoir récupérer une Case quelconque (à partir de son numéro ou de sa position) parmi toutes les Cases. En conséquence, l'objet Plateau (qui possède une relation de composition) dispose des informations nécessaires pour faire cela.