

Technologies objet



Design patterns

par la pratique

- A. Shalloway
- J.R. Trott

EYROLLES

2-005-627-1

Alan Shalloway - James Trott

Table des matières

Design patterns par la pratique



EYROLLES

Table des matières

Préface	XV
----------------------	----

SECTION I

Introduction au développement de logiciels orientés objet	1
--	---

CHAPITRE 1

Le paradigme orienté objet	3
---	---

Au début était l'analyse fonctionnelle	4
---	---

Les modifications du code sont source d'erreur	4
--	---

Le problème des spécifications	5
---	---

Les spécifications évoluent sans cesse	5
--	---

Gestion des modifications par découpage en modules	6
--	---

Problèmes de la décomposition fonctionnelle	6
---	---

Faible cohésion, fort couplage	6
--------------------------------------	---

Bogues, effets secondaires	7
----------------------------------	---

Inconvénients de la décomposition fonctionnelle	7
---	---

Gestion des changements de spécifications	8
--	---

Petite étude du comportement humain	8
---	---

Délégation de responsabilités et modifications	9
--	---

Les différents niveaux du développement	9
---	---

L'efficacité des niveaux	10
--------------------------------	----

Le paradigme orienté objet.	10
--	----

Représentation objet et délégation de responsabilités	10
---	----

Comment voir les objets	11
-------------------------------	----

Interface publique	11
--------------------------	----

Organisation des objets en classes	12
--	----

Les objets sont des instances de classes	12
--	----

Utilisation de l'approche orientée objet dans notre exemple	12
---	----

Nécessité d'un type abstrait	13
Vertus des classes abstraites	14
Visibilité	14
Encapsulation	15
Polymorphisme	15
La programmation orientée objet en pratique	16
Avantages du nouveau style	17
Retour à l'encapsulation	17
Les mérites de l'encapsulation	18
Méthodes particulières	18
 CHAPITRE 2	
UML, le langage de modélisation unifié	21
Définition du langage UML	21
Avantages présentés par UML	22
Diagrammes de classes	23
Description des classes	23
Relations entre les classes	24
 SECTION II	
Limites de la conception orientée objet traditionnelle	31
 CHAPITRE 3	
Étude de cas illustrant la nécessité d'un code flexible	33
Extraire des données d'un système de CFAO	33
Terminologie spécifique au domaine	34
Terminologie du travail des métaux	34
Terminologie de la CFAO	35
Contraintes et solutions	37
Système expert et évolution du système de CFAO	37
Diagramme de classes global	38
Deux versions du système de CFAO	39
 CHAPITRE 4	
Solution orientée objet standard	41
Solutions au cas par cas	41
Diagrammes de la première solution	41

Extraits du code Java correspondant au diagramme	45
Extraits de code équivalents en C++	48

SECTION III

Design Patterns	51
CHAPITRE 5	
Introduction aux design patterns	53
Les design patterns comme produits de l'architecture et de l'anthropologie	54
Appliquer les concepts définis pour l'architecture aux design patterns informatiques	56
Intérêt des design patterns	57
Réutiliser les solutions	57
Approche plus globale	58
Améliorer la communication et l'apprentissage	60
Souplesse du logiciel	60
Principes de base orientés objet	60
Alternative aux hiérarchies d'héritage démesurées	61
CHAPITRE 6	
Le pattern Façade	63
Présentation du pattern Façade	63
Le pattern Façade en contexte	64
Le pattern Façade : caractéristiques	65
Le pattern Façade : notes pratiques	65
Rôle du pattern Façade dans notre exemple de CFAO	67
En bref	67
CHAPITRE 7	
Le pattern Adaptateur	69
Présentation du pattern Adaptateur	69
Le pattern Adaptateur en contexte	70
L'objet client ignore les détails	70
Polymorphisme et classes dérivées	71
Définition de l'interface et implémentation des classes dérivées	71
Le pattern Adaptateur : caractéristiques	75

Le pattern Adaptateur : notes pratiques	75
Au-delà de l'encapsulation	75
L'interface des classes existantes n'est plus une contrainte	76
Adaptateur d'objet et Adaptateur de classe	76
Comparaison entre le pattern Adaptateur et le pattern Façade	76
Rôle du pattern Adaptateur dans notre exemple de CFAO	77
Exemple de code en C++	78
 CHAPITRE 8	
Au-delà de l'approche traditionnelle	79
Objets : approche traditionnelle et nouvelle approche	80
Approche traditionnelle : des données associées à des méthodes	80
Nouvelle approche : des choses avec des responsabilités	80
Encapsulation : approche traditionnelle et nouvelle approche	81
Définition exhaustive	81
Plusieurs niveaux d'encapsulation	81
Recherche et encapsulation des variations	83
L'héritage et les design patterns	83
Variations des données ou des comportements	83
Gestion des variations du comportement grâce aux objets	84
Comparaison des deux solutions	85
Communalité, variabilité et classes abstraites	86
 CHAPITRE 9	
Le pattern Pont	89
Présentation du pattern Pont (« Bridge »)	89
Le pattern Pont en contexte	90
Données de base du problème : comment dessiner des formes ?	90
Utilisation judicieuse de l'héritage	91
Implémentation simple d'une variation	92
Tentative de conception	95
Remarque sur l'utilisation des design patterns	100
Principes d'élaboration du pattern Pont	100
Analyse de la communalité et de la variabilité	101
Stratégies de gestion des variations	101
Approche conceptuelle du pattern Pont	109
Le pattern Pont : caractéristiques	110
Le pattern Pont : notes pratiques	110

Design pattern composé	110
Instanciation des objets	111
Java plutôt que C++	111
Une solution presque parfaite	111
Le refactoring	112
Relations entre les classes	112
Rappel sur les principes orientés objet	113
Exemples de code en C++	114

CHAPITRE 10

Le modèle Fabrique abstraite	119
Présentation du pattern Fabrique abstraite	119
Le pattern Fabrique abstraite : élaboration	120
Exemple : sélection de pilotes de périphériques en fonction de la capacité d'une machine	120
Première solution : utiliser une instruction switch pour sélectionner le pilote	120
Deuxième solution : utiliser l'héritage	122
Troisième solution : remplacer les instructions switch par l'abstraction ..	123
Objet fabrique (Factory)	124
Le pattern Fabrique abstraite : implémentation	127
Le pattern Fabrique abstraite : caractéristiques	129
Le pattern Fabrique abstraite : notes pratiques	129
Avantages du pattern Fabrique abstraite	130
Familles d'objets	131
Deux variantes : les fichiers de configuration et la classe Class en Java ..	131
Le pattern Fabrique abstraite et les adaptateurs	132
Rôle du pattern Fabrique abstraite dans notre problème CFAO	132
Exemples de code en C++	132

PARTIE IV

Synthèse : raisonner en termes de patterns	135
---	------------

CHAPITRE 11

Comment les experts conçoivent-ils leurs projets ?	137
De l'architecture aux logiciels	138
Construire en assemblant des éléments	138

Qui dit bonne conception, dit approche globale	140
Comment procéder ?	141
CHAPITRE 12	
Résolution du problème de CFAO à l'aide de patterns	145
Rappel du problème de CFAO	145
Raisonnement en termes de patterns	146
Étape 1 : identification des patterns	147
Étape 2a : recherche du pattern de contexte	147
Étape 2b : définition du problème sous forme d'un tout	151
Étape 2c : identification de patterns supplémentaires	155
Étape 2d : le pattern Façade	155
Étape 2d-bis : le pattern Adaptateur	156
Étape 2d-ter : le pattern Fabrique abstraite	156
Étape 3 : ajout des détails et affectation des responsabilités	157
Comparaison avec la solution précédente	157
CHAPITRE 13	
Principes et stratégies qui sous-tendent les design patterns	159
Le principe ouvert-fermé	160
Le principe de conception à partir du contexte	160
Dans le cadre du pattern Pont	160
Dans le cadre du pattern Fabrique abstraite	161
Dans le cadre du pattern Adaptateur	162
Dans le cadre du pattern Façade	163
Le principe d'encapsulation des variations	163
SECTION V	
Gestion des variations à l'aide des design patterns	165
CHAPITRE 14	
Le pattern Stratégie	167
Approche de la gestion de nouvelles spécifications	167
Spécifications initiales de l'étude de cas	169
Gestion de nouvelles spécifications	170
Le pattern Stratégie	173
Le pattern Stratégie : caractéristiques	174

Le pattern Stratégie : notes pratiques	175
CHAPITRE 15	
Le pattern Décorateur	177
Une nouvelle spécification	177
Le pattern Décorateur	179
Application du pattern Décorateur à l'étude de cas	181
Autre utilisation du pattern Décorateur : flux d'entrée/sortie	184
Le pattern Décorateur : caractéristiques	186
Le pattern Décorateur : notes pratiques	186
Exemples de code en C++	187
CHAPITRE 16	
Les patterns Singleton et Verrouillage à double tour	191
Présentation du pattern Singleton	191
Application du pattern Singleton à l'étude de cas	192
Le pattern Singleton : caractéristiques	193
Une variante : le pattern Verrouillage à double tour	193
Les patterns Singleton et Verrouillage à double tour : notes pratiques	195
Exemples de code en C++	196
CHAPITRE 17	
Le pattern Observateur	197
Catégories de patterns	197
Spécifications supplémentaires de l'étude de cas	198
Le pattern Observateur	200
Application de l'observateur à l'étude de cas	200
Étape 1 : les observateurs doivent se comporter de la même façon	200
Étape 2 : les observateurs doivent s'enregistrer	201
Étape 3 : avertir les observateurs de l'événement	201
Étape 4 : obtenir les informations du sujet	201
Application	201
Le pattern Observateur : caractéristiques	205
Le pattern Observateur : notes pratiques	205
Résumé des principes orientés objet dans le cadre du pattern Observateur	207
Exemple de code en C++	207

CHAPITRE 18

Le pattern Patron de méthode	211
Spécifications supplémentaires de l'étude de cas	211
Le pattern Patron de méthode (Template method)	212
Application du patron de méthode à l'étude de cas	212
Le pattern Patron de méthode : caractéristiques	214
Le pattern Patron de méthode : notes pratiques	214

CHAPITRE 19

Le pattern Fabrication	217
Spécifications supplémentaires de l'étude de cas	217
Le pattern Fabrication (Factory Method)	219
Le pattern Fabrication : caractéristiques	219
Le pattern Fabrication : notes pratiques	220

CHAPITRE 20

La matrice d'analyse	221
Les variations dans la pratique	221
Les variations dans la pratique : un système de commerce électronique international	222
Étape 1 : identification et organisation des caractéristiques importantes ..	223
Étape 2 : traitement des autres cas et extension de la matrice en conséquence	224
Étape 3 : extension de la matrice d'analyse à l'aide de nouveaux concepts	225
Étape 4 : utilisation des lignes pour identifier les règles	226
Étape 5 : utilisation des colonnes pour identifier l'implémentation	226
Étape 6 : identification des design patterns à partir des lignes de l'analyse	227
Étape 7 : identification des design patterns à partir des colonnes de l'analyse	228
Étape 8 : mise au point d'une conception globale	228
Notes pratiques	228

SECTION VI

Conclusion et poursuite de la découverte	231
CHAPITRE 21	
Rappel des principes des design patterns	233
Résumé des principes orientés objet	234
Le rôle des design patterns dans l'encapsulation des implémentations	234
Le rôle de l'analyse de la communalité/variabilité dans l'implémentation des design patterns	235
Décomposition du domaine d'un problème en responsabilités	235
Relations dans le cadre d'un pattern	236
Design patterns et conception contextuelle	236
Notes pratiques	237
Un dernier conseil	238
CHAPITRE 22	
Bibliographie	239
Site web du livre anglais	240
Références et ouvrages conseillés sur les design patterns et l'orientation objet	240
Références et ouvrages conseillés sur la programmation Java	241
Ouvrage conseillé sur le thème de la programmation C++	242
Ouvrage conseillé sur la programmation COBOL	242
Références et ouvrages conseillés sur la programmation extrême (eXtreme Programming)	242
Ouvrage conseillé sur le thème de la programmation en général	243
Favoris des auteurs	243
Recommandations d'Alan Shalloway	243
Recommandations de Jim Trott	244
Ouvrages en français conseillés sur les design patterns et la conception objet	245
Index	247