

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

---

|                      |    |
|----------------------|----|
| <b>Préface</b> ..... | XV |
|----------------------|----|

## SECTION I

---

|   |   |
|---|---|
| <b>Introduction au développement de logiciels orientés objet</b> .. | 1 |
|---|---|

### CHAPITRE 1

|   |   |
|---|---|
| <b>Le paradigme orienté objet</b> ..... | 3 |
|---|---|

|   |   |
|---|---|
| <b>Au début était l'analyse fonctionnelle</b> ..... | 4 |
|---|---|

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

|   |   |
|---|---|
| <b>Le problème des spécifications</b> ..... | 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 |
|---|---|

|  |   |
|--|---|
| <b>Gestion des changements de spécifications</b> ..... | 8 |
|--|---|

|   |   |
|---|---|
| Petite étude du comportement humain ..... | 8 |
|---|---|

|  |   |
|--|---|
| Délégation de responsabilités et modifications ..... | 9 |
|--|---|

|   |   |
|---|---|
| Les différents niveaux du développement ..... | 9 |
|---|---|

|                                |    |
|--------------------------------|----|
| L'efficacité des niveaux ..... | 10 |
|--------------------------------|----|

|  |    |
|--|----|
| <b>Le paradigme orienté objet.</b> ..... | 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        |
| <b>La programmation orientée objet en pratique .....</b>             | <b>16</b> |
| Avantages du nouveau style .....                                     | 17        |
| Retour à l'encapsulation .....                                       | 17        |
| Les mérites de l'encapsulation .....                                 | 18        |
| <b>Méthodes particulières .....</b>                                  | <b>18</b> |
| <br>CHAPITRE 2   |           |
| <b>UML, le langage de modélisation unifié .....</b>                  | <b>21</b> |
| Définition du langage UML .....                                      | 21        |
| Avantages présentés par UML .....                                    | 22        |
| <b>Diagrammes de classes .....</b>                                   | <b>23</b> |
| Description des classes .....  | 23        |
| Relations entre les classes .....                                    | 24        |
| <br>SECTION II   |           |
| <b>Limites de la conception orientée objet traditionnelle .....</b>  | <b>31</b> |
| <br>CHAPITRE 3   |           |
| <b>Étude de cas illustrant la nécessité d'un code flexible .....</b> | <b>33</b> |
| Extraire des données d'un système de CFAO .....                      | 33        |
| <b>Terminologie spécifique au domaine .....</b>                      | <b>34</b> |
| Terminologie du travail des métaux .....                             | 34        |
| Terminologie de la CFAO .....  | 35        |
| <b>Contraintes et solutions .....</b>                                | <b>37</b> |
| Système expert et évolution du système de CFAO .....                 | 37        |
| Diagramme de classes global .....                                    | 38        |
| Deux versions du système de CFAO .....                               | 39        |
| <br>CHAPITRE 4   |           |
| <b>Solution orientée objet standard .....</b>                        | <b>41</b> |
| 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

|   |    |
|---|----|
| <b>Design Patterns</b> .....  | 51 |
| CHAPITRE 5  |    |
| <b>Introduction aux design patterns</b> .....   | 53 |
| Les design patterns comme produits de l'architecture<br>et de l'anthropologie .....           | 54 |
| Appliquer les concepts définis pour l'architecture aux design patterns<br>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  |    |
| <b>Le pattern Façade</b> .....  | 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  |    |
| <b>Le pattern Adaptateur</b> .....  | 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 |

|   |     |
|---|-----|
| <b>Le pattern Adaptateur : notes pratiques</b> .....                      | 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  |
| <b>Rôle du pattern Adaptateur dans notre exemple de CFAO</b> .....        | 77  |
| <b>Exemple de code en C++</b> .....                                       | 78  |
| <br>CHAPITRE 8  |     |
| <b>Au-delà de l'approche traditionnelle</b> .....                         | 79  |
| <b>Objets : approche traditionnelle et nouvelle approche</b> .....        | 80  |
| Approche traditionnelle : des données associées à des méthodes .....      | 80  |
| Nouvelle approche : des choses avec des responsabilités .....             | 80  |
| <b>Encapsulation : approche traditionnelle et nouvelle approche</b> ..... | 81  |
| Définition exhaustive .....   | 81  |
| Plusieurs niveaux d'encapsulation .....                                   | 81  |
| <b>Recherche et encapsulation des variations</b> .....                    | 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  |
| <br>CHAPITRE 9  |     |
| <b>Le pattern Pont</b> .....  | 89  |
| <b>Présentation du pattern Pont (« Bridge »)</b> .....                    | 89  |
| <b>Le pattern Pont en contexte</b> .....                                  | 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  |
| <b>Remarque sur l'utilisation des design patterns</b> .....               | 100 |
| <b>Principes d'élaboration du pattern Pont</b> .....                      | 100 |
| Analyse de la communalité et de la variabilité .....                      | 101 |
| Stratégies de gestion des variations .....                                | 101 |
| <b>Approche conceptuelle du pattern Pont</b> .....                        | 109 |
| <b>Le pattern Pont : caractéristiques</b> .....                           | 110 |
| <b>Le pattern Pont : notes pratiques</b> .....                            | 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        |
| <b>Rappel sur les principes orientés objet .....</b> | <b>113</b> |
| <b>Exemples de code en C++ .....</b>                 | <b>114</b> |

## CHAPITRE 10

|   |            |
|---|------------|
| <b>Le modèle Fabrique abstraite .....</b>   | <b>119</b> |
| Présentation du pattern Fabrique abstraite .....  | 119        |
| <b>Le pattern Fabrique abstraite : élaboration .....</b>  | <b>120</b> |
| Exemple : sélection de pilotes de périphériques en fonction<br>de la capacité d'une machine ..... | 120        |
| Première solution : utiliser une instruction switch pour<br>sélectionner le pilote .....          | 120        |
| Deuxième solution : utiliser l'héritage .....   | 122        |
| Troisième solution : remplacer les instructions switch par l'abstraction ..                       | 123        |
| Objet fabrique (Factory) .....  | 124        |
| <b>Le pattern Fabrique abstraite : implémentation .....</b>                                       | <b>127</b> |
| <b>Le pattern Fabrique abstraite : caractéristiques .....</b>                                     | <b>129</b> |
| <b>Le pattern Fabrique abstraite : notes pratiques .....</b>                                      | <b>129</b> |
| 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        |
| <b>Rôle du pattern Fabrique abstraite dans notre problème CFAO ....</b>                           | <b>132</b> |
| <b>Exemples de code en C++ .....</b>  | <b>132</b> |

## PARTIE IV

|   |            |
|---|------------|
| <b>Synthèse : raisonner en termes de patterns .....</b> | <b>135</b> |
|---|------------|

## CHAPITRE 11

|   |            |
|---|------------|
| <b>Comment les experts conçoivent-ils leurs projets ? .....</b> | <b>137</b> |
| 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

|  |            |
|--|------------|
| <b>Résolution du problème de CFAO à l'aide de patterns .....</b>     | <b>145</b> |
| <b>Rappel du problème de CFAO .....</b>                              | <b>145</b> |
| <b>Raisonnement en termes de patterns .....</b>                      | <b>146</b> |
| 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        |
| <b>Comparaison avec la solution précédente .....</b>                 | <b>157</b> |

## CHAPITRE 13

|   |            |
|---|------------|
| <b>Principes et stratégies qui sous-tendent les design patterns .....</b> | <b>159</b> |
| <b>Le principe ouvert-fermé .....</b>                                     | <b>160</b> |
| <b>Le principe de conception à partir du contexte .....</b>               | <b>160</b> |
| 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        |
| <b>Le principe d'encapsulation des variations .....</b>                   | <b>163</b> |

## SECTION V

|  |            |
|--|------------|
| <b>Gestion des variations à l'aide des design patterns .....</b> | <b>165</b> |
|--|------------|

## CHAPITRE 14

|  |            |
|--|------------|
| <b>Le pattern Stratégie .....</b>                        | <b>167</b> |
| 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   |     |
| <b>Le pattern Décorateur</b> .....  | 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   |     |
| <b>Les patterns Singleton et Verrouillage à double tour</b> .....                 | 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   |     |
| <b>Le pattern Observateur</b> .....   | 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<br>du pattern Observateur ..... | 207 |
| Exemple de code en C++ .....  | 207 |

## CHAPITRE 18

|   |     |
|---|-----|
| <b>Le pattern Patron de méthode</b> .....               | 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

|  |     |
|--|-----|
| <b>Le pattern Fabrication</b> .....                    | 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

|  |     |
|--|-----|
| <b>La matrice d'analyse</b> .....  | 221 |
| Les variations dans la pratique .....  | 221 |
| Les variations dans la pratique : un système de commerce<br>électronique international ..... | 222 |
| Étape 1 : identification et organisation des caractéristiques importantes ..                 | 223 |
| Étape 2 : traitement des autres cas et extension de la matrice<br>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<br>de l'analyse .....     | 228 |
| Étape 8 : mise au point d'une conception globale .....                                       | 228 |
| Notes pratiques .....  | 228 |

## SECTION VI

|   |     |
|---|-----|
| <b>Conclusion et poursuite de la découverte</b> .....   | 231 |
| CHAPITRE 21   |     |
| <b>Rappel des principes des design patterns</b> .....   | 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<br>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   |     |
| <b>Bibliographie</b> .....  | 239 |
| Site web du livre anglais .....   | 240 |
| Références et ouvrages conseillés sur les design patterns<br>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<br>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<br>et la conception objet .....               | 245 |
| <b>Index</b> .....  | 247 |