



Craig Larman

UML 2 et les design patterns

Analyse et conception orientées objet
et développement itératif

3^e édition

PEARSON
Education

Table des matières

| | |
|--|--------|
| Préface | XXVIII |
| Préface à l'édition française | XXIX |
| Avant-propos | 1 |
| Public visé | 2 |
| Prérequis | 2 |
| Exemples en Java | 2 |
| Plan de l'ouvrage | 2 |
| À propos de l'auteur | 3 |
| Contact | 3 |
| Pourquoi une nouvelle édition ? | 3 |
| Remerciements | 4 |
| Conventions typographiques | 5 |

Partie I

Introduction

| | |
|--|----|
| 1. Analyse et conception orientées objet | 9 |
| 1.1 Objectifs de l'ouvrage | 9 |
| UML et pensée « objet » | 10 |
| Conception orientée objet : principes et patterns | 10 |
| Études de cas | 11 |
| Cas d'utilisation | 11 |
| Développement itératif, modélisation agile et UP agile | 11 |
| Compétences importantes | 12 |
| 1.2 Quel est l'objectif d'apprentissage le plus important ? | 12 |
| 1.3 Qu'est-ce que l'analyse et la conception ? | 12 |
| 1.4 Qu'est-ce que l'analyse et la conception orientées objet ? | 13 |

| | | |
|-----------|--|-----------|
| 1.5 | Vue globale par l'exemple | 13 |
| | Définir les cas d'utilisation | 14 |
| | Définir un modèle du domaine | 14 |
| | Affecter des responsabilités aux objets et tracer des diagrammes d'interaction | 15 |
| | Définir des diagrammes de classes de conception | 16 |
| | En résumé | 16 |
| 1.6 | Qu'est-ce qu'UML ? | 16 |
| | Trois façons d'appliquer UML | 17 |
| | Trois perspectives pour l'application d'UML | 18 |
| | Signification de « classe » dans les différentes perspectives | 19 |
| | UML 1 et UML 2 | 20 |
| 1.7 | L'intérêt de la modélisation graphique | 20 |
| 1.8 | Historique de l'A/COO | 20 |
| 1.9 | Ressources complémentaires | 22 |
| 2. | Développement itératif, évolutif et agile | 23 |
| 2.1 | Le Processus Unifié | 24 |
| | Et si je n'ai pas besoin du Processus Unifié ? | 25 |
| 2.2 | Qu'est-ce que le développement itératif et évolutif ? | 25 |
| | Gérer le changement dans un projet itératif | 26 |
| | Les avantages du développement itératif | 28 |
| | La durée d'une itération | 28 |
| 2.3 | Le cycle de vie en cascade | 29 |
| | Principe : ne laissez pas la pensée en cascade envahir un projet UP | 30 |
| | Pourquoi l'approche en cascade est-elle sujette aux échecs ? | 30 |
| | Nécessité du feed-back et de l'adaptation | 31 |
| 2.4 | Comment pratiquer l'analyse et la conception itératives et évolutives ? | 31 |
| 2.5 | La planification itérative pilotée par les risques et pilotée par le client | 34 |
| 2.6 | Les méthodes et les attitudes agiles | 34 |
| | Le manifeste agile et ses principes | 35 |
| 2.7 | La modélisation agile | 36 |
| | La modélisation agile dans cet ouvrage : pourquoi des instantanés des esquisses UML ? | 37 |
| 2.8 | Qu'est-ce que UP agile ? | 38 |
| 2.9 | Le Processus Unifié comprend-il d'autres bonnes pratiques ? | 39 |

| | | |
|-----------|---|-----------|
| 1.1 | Les phases du Processus Unifié | 40 |
| 1.2 | Les disciplines du Processus Unifié | 40 |
| 1.3 | Les relations entre disciplines et phases | 42 |
| 1.4 | Structure de l'ouvrage, phases et disciplines | 42 |
| 2.12 | Personnalisation du processus : le Cas de Développement UP | 43 |
| | Le Processus Unifié comprend-il des pratiques ou des artefacts optionnels ? | 43 |
| | Définition : qu'est-ce que le Cas de développement ? | 44 |
| 2.13 | Bien comprendre le Processus Unifié | 44 |
| 2.14 | Historique | 46 |
| 2.15 | Ressources complémentaires | 46 |
| | Ressources Web | 47 |
| 3. | Les études de cas | 49 |
| 3.1 | Les sujets abordés et non abordés dans les études de cas | 49 |
| 3.2 | Principe de l'ouvrage : développement itératif et apprentissage itératif | 50 |
| 3.3 | Étude de cas 1 : le système NextGen | 51 |
| 3.4 | Étude de cas 2 : le jeu de Monopoly | 51 |

Partie II Inception

| | | |
|-----------|--|-----------|
| 4. | Inception et définition des besoins | 55 |
| 4.1 | La phase d'inception | 55 |
| | L'inception : une analogie | 56 |
| 4.2 | Quelle durée accorder à l'inception ? | 57 |
| 4.3 | Les artefacts de l'inception | 57 |
| | L'inception représente-t-elle une documentation importante ? ... | 58 |
| 4.4 | Quelle proportion d'UML en phase d'inception ? | 59 |
| 5. | Les besoins et leur évolution | 61 |
| 5.1 | Les besoins : définition | 61 |
| 5.2 | Approche évolutive et approche en cascade | 62 |
| 5.3 | Déterminer les besoins | 64 |
| 5.4 | Les types et les catégories de besoins | 64 |

| | | |
|-----------|--|-----------|
| 5.5 | Comment les besoins sont-ils organisés dans les artefacts du Processus Unifié ? | 65 |
| | Quel est le format correct de ces artefacts ? | 66 |
| 5.6 | L'ouvrage contient-il des exemples d'artefacts ? | 66 |
| 5.7 | Ressources complémentaires | 66 |
| 6. | Cas d'utilisation | 69 |
| 6.1 | Exemple de cas d'utilisation | 70 |
| 6.2 | Définitions : acteurs, scénarios et cas d'utilisation | 71 |
| 6.3 | Cas d'utilisation et Modèle de Cas d'Utilisation | 72 |
| 6.4 | Motivation : pourquoi des cas d'utilisation ? | 72 |
| 6.5 | Définition : les cas d'utilisation sont-ils des besoins fonctionnels ? | 73 |
| 6.6 | Définition : quels sont les trois types d'acteurs ? | 73 |
| 6.7 | Les trois formats des cas d'utilisation | 74 |
| 6.8 | Exemple de cas d'utilisation détaillé : Traiter une vente | 74 |
| 6.9 | Que signifient les sections ? | 81 |
| | Éléments de préface | 81 |
| | Liste des parties prenantes et des intérêts | 82 |
| | Préconditions et garanties en cas de succès (postconditions) | 83 |
| | Scénario principal (succès) et étapes | 83 |
| | Extensions (ou scénarios alternatifs) | 84 |
| | Exécution d'un autre scénario | 86 |
| | Spécifications particulières | 86 |
| | Liste de variantes de données et de technologies | 87 |
| 6.10 | Notation : existe-t-il d'autres formats ? Une variante sur deux colonnes | 88 |
| | Quel est le meilleur format ? | 89 |
| 6.11 | Principe : rédigez les cas d'utilisation en style essentiel, indépendamment des interfaces utilisateur | 89 |
| | Le cas de l'empreinte digitale | 89 |
| | Le style « essentiel » | 89 |
| | Exemples à comparer | 90 |
| 6.12 | Principe : rédigez avec concision | 91 |
| 6.13 | Principe : rédigez des cas d'utilisation en « boîte noire » | 91 |
| 6.14 | Principe : adoptez une perspective centrée sur les acteurs et les buts | 91 |

| | | |
|-----------|---|------------|
| 6.15 | Comment découvrir les cas d'utilisation ? | 92 |
| | Étape 1 : choisir les limites du système | 92 |
| | Étapes 2 et 3 : identifier les acteurs principaux et auxiliaires | 93 |
| | Pourquoi poser des questions sur les buts des acteurs plutôt que sur les cas d'utilisation ? | 94 |
| | Étape 4 : définir les cas d'utilisation | 96 |
| 6.16 | Principe : quels sont les tests qui permettent d'identifier des cas d'utilisation valides ? | 96 |
| | Le test du patron | 97 |
| | Le test PME | 97 |
| | Le test de la taille | 98 |
| | Exemple : appliquer les tests | 98 |
| 6.17 | Application d'UML : diagrammes de cas d'utilisation | 99 |
| | Suggestions pour le traçage des diagrammes | 100 |
| | Principe : attention aux excès de diagrammes | 101 |
| 6.18 | Application d'UML : diagrammes d'activités | 101 |
| 6.19 | Autres avantages des cas d'utilisation : les besoins dans leur contexte | 101 |
| | Les listes de fonctionnalités système de haut niveau sont acceptables | 102 |
| | Quand les listes de fonctionnalités sont-elles préférables aux cas d'utilisation ? | 102 |
| 6.20 | Exemple : le jeu de Monopoly | 103 |
| 6.21 | Les cas d'utilisation dans les méthodes itératives | 104 |
| | Faire évoluer les cas d'utilisation et les autres spécifications à travers les itérations | 104 |
| | Créer les différents artefacts du Processus Unifié (dont les cas d'utilisation) | 107 |
| | Rédiger les cas d'utilisation en phase d'inception | 107 |
| | Rédiger les cas d'utilisation en phase d'élaboration | 108 |
| | Rédiger les cas d'utilisation en phase de construction | 108 |
| | Étude de cas : les cas d'utilisation en phase d'inception de NextGen . | 109 |
| 6.22 | Historique | 109 |
| 6.23 | Ressources complémentaires | 109 |
| 7. | Autres besoins | 111 |
| | Autres artefacts liés aux besoins | 111 |
| 7.1 | Les exemples sont-ils complets ? | 112 |

| | | |
|------|--|-----|
| 7.2 | Principe : doit-on analyser ces autres besoins en phase d'inception ? | 112 |
| | Des spécifications fiables : est-ce une gageure ? | 112 |
| 7.3 | Principe : doit-on placer ces artefacts en ligne sur le site Web du projet ? | 113 |
| 7.4 | Exemple issu du système NextGen : Spécifications Supplémentaires (partielles) | 113 |
| 7.5 | Commentaire : Spécifications Supplémentaires | 117 |
| | Attributs de qualité | 118 |
| | Les fonctionnalités ont-elles leur place dans les Spécifications Supplémentaires ou dans les cas d'utilisation ? | 118 |
| | Règles du domaine (de gestion) spécifiques à l'application | 118 |
| | Informations relatives aux domaines d'intérêt | 118 |
| 7.6 | Exemple NextGen : Vision (partielle) | 119 |
| 7.7 | Commentaire : Vision | 123 |
| | Principe : faut-il faire figurer les autres spécifications dans la Vision ? | 125 |
| | Principe : faut-il rédiger d'abord la Vision ou les cas d'utilisation ? | 125 |
| 7.8 | Exemple issu de NextGen : Glossaire (partiel) | 126 |
| 7.9 | Commentaire : Glossaire (dictionnaire de données) | 127 |
| | Utilisation du Glossaire comme dictionnaire de données | 127 |
| | Principe : peut-on utiliser le Glossaire pour les termes composites ? | 127 |
| 7.10 | Exemple issu de NextGen : règles de gestion (règles du domaine) | 128 |
| 7.11 | Commentaire : règles du domaine | 129 |
| 7.12 | Les besoins et leur évolution dans les méthodes itératives | 129 |
| | Inception | 130 |
| | Élaboration | 130 |
| | Construction | 131 |
| 7.13 | Ressources complémentaires | 131 |

Partie III

Élaboration de l'itération 1

| | | |
|-----------------------------------|--|-----|
| 8. Itération 1 – les bases | 135 | |
| 8.1 | Itération 1 : compétences A/COO fondamentales | 135 |
| | NextGen | 136 |
| | Monopoly | 136 |
| | Les spécifications sont implémentées progressivement | 137 |

| | | |
|-----------|---|------------|
| | Développement incrémental du même cas d'utilisation entre itérations | 137 |
| 8.2 | Processus : inception et élaboration | 138 |
| | Que s'est-il passé lors de la phase d'inception ? | 138 |
| | Passage à l'élaboration | 139 |
| | Quels artefacts peut-on commencer en phase d'élaboration ? | 140 |
| 8.3 | Planification de l'itération suivante | 141 |
| 9. | Modèles du domaine : visualisation des concepts | 143 |
| 9.1 | Exemple de modèle du domaine | 144 |
| 9.2 | Qu'est-ce qu'un modèle du domaine ? | 145 |
| | Définition : pourquoi qualifier le Modèle du Domaine de dictionnaire visuel ? | 146 |
| | Définition : un modèle du domaine est-il une image des objets métier logiciels | 147 |
| | Définition : quelles sont les deux significations traditionnelles de « Modèle du Domaine » ? | 148 |
| | Définition : qu'est-ce qu'une classe conceptuelle ? | 148 |
| | Définition : un modèle du domaine est-il un modèle de données ? | 149 |
| 9.3 | Motivation : pourquoi créer un Modèle du Domaine ? | 149 |
| | Motivation : réduire le décalage des représentations avec la modélisation objet | 150 |
| 9.4 | Principe : comment créer un modèle du domaine ? | 151 |
| 9.5 | Comment identifier les classes conceptuelles ? | 151 |
| | Méthode 1 : réutiliser ou modifier des modèles existants. | 151 |
| | Méthode 2 : utiliser une liste de catégories | 152 |
| | Méthode 3 : identifier les classes conceptuelles à l'aide de groupes nominaux | 153 |
| 9.6 | Exemple : identifier et représenter les classes conceptuelles | 155 |
| | Étude de cas : le domaine des caisses enregistreuses | 155 |
| | Étude de cas : le domaine du Monopoly | 156 |
| 9.7 | Principe : modélisation agile – tracer un diagramme de classes en mode esquisse | 156 |
| 9.8 | Principe : modélisation agile – utiliser un outil pour maintenir le modèle ? | 156 |
| 9.9 | Principe : objets de type « rapport » – inclure le Reçu dans le modèle ? | 157 |

| | | |
|------|---|-----|
| 9.10 | Principe : penser comme un cartographe – employer les termes du domaine | 157 |
| 9.11 | Principe : comment modéliser le monde irréel ? | 158 |
| 9.12 | Principe : une erreur courante – confondre les attributs et les classes | 158 |
| 9.13 | Principe : comment modéliser avec des classes de « description » ? | 159 |
| | Motivation : pourquoi utiliser des classes de « description » ? | 159 |
| | Principe : quand les classes de description sont-elles utiles ? | 160 |
| | Exemple : descriptions dans le domaine des compagnies aériennes | 160 |
| 9.14 | Associations | 162 |
| | Principe : quand représenter une association ? | 162 |
| | Principe : pourquoi éviter d'ajouter trop d'associations ? | 163 |
| | Perspectives : toutes les associations seront-elles implémentées ? | 163 |
| | Application d'UML : notation des associations | 163 |
| | Principe : comment nommer une association en UML ? | 164 |
| | Application d'UML : rôles | 165 |
| | Application d'UML : multiplicité | 165 |
| | Application d'UML : associations multiples entre deux classes ... | 166 |
| | Principe : comment découvrir des associations avec une liste d'associations courantes ? | 167 |
| 9.15 | Les associations dans le modèle du domaine | 168 |
| | Étude de cas : NextGen | 168 |
| | Étude de cas : Monopoly | 169 |
| 9.16 | Attributs | 169 |
| | Principe : quand représenter les attributs ? | 169 |
| | Application d'UML : notation des attributs | 170 |
| | Principe : quels sont les types d'attributs valides ? | 172 |
| | Types de données | 173 |
| | Principe : quand définir de nouveaux types ? | 173 |
| | Application d'UML : où représenter les classes types de données ? | 175 |
| | Principe : pas d'attributs pour représenter les clés étrangères | 175 |
| | Principe : modélisation des quantités et des unités | 176 |
| 9.17 | Exemple : les attributs dans le modèle du domaine | 176 |
| | Étude de cas : NextGen | 176 |
| | Étude de cas : Monopoly | 178 |
| 9.18 | Conclusion : le modèle du domaine est-il correct ? | 178 |

| | | |
|------------|---|------------|
| 9.19 | Processus : modélisation du domaine itérative et évolutive | 178 |
| | Le Modèle du Domaine dans le Processus Unifié | 179 |
| | Inception | 179 |
| | Élaboration | 180 |
| | Modèle d'Objets Métier et Modèle du Domaine | 180 |
| 9.20 | Ressources complémentaires | 180 |
| 10. | Diagrammes de séquence système (DSS) | 181 |
| 10.1 | Qu'est-ce qu'un diagramme de séquence système ? | 183 |
| 10.2 | Motivation : pourquoi tracer un DSS ? | 184 |
| 10.3 | Application d'UML : diagrammes de séquence | 184 |
| | Boucles dans les diagrammes de séquence | 184 |
| 10.4 | La relation entre les DSS et les cas d'utilisation | 184 |
| | Application d'UML : faut-il faire figurer le texte des cas d'utilisation dans le DSS ? | 185 |
| 10.5 | Nommer les opérations et les événements système | 185 |
| 10.6 | Modéliser des DSS impliquant des systèmes externes | 186 |
| 10.7 | Quelles informations du DSS placer dans le Glossaire ? | 186 |
| 10.8 | Exemple : le DSS du Monopoly | 186 |
| 10.9 | Processus : DSS itératifs et évolutifs | 187 |
| | Les DSS dans le Processus Unifié | 187 |
| | Phases du Processus Unifié | 187 |
| 10.10 | Historique et ressources complémentaires | 188 |
| 11. | Contrats d'opération | 189 |
| 11.1 | Exemple de contrat d'opération | 190 |
| 11.2 | Définition : quelles sont les sections d'un contrat ? | 191 |
| 11.3 | Définition : qu'est-ce qu'une opération système ? | 191 |
| 11.4 | Définition : postconditions | 192 |
| | Comment les postconditions sont-elles liées au Modèle du Domaine ? | 193 |
| | Motivation : pourquoi des postconditions ? | 193 |
| | Principe : comment rédiger une postcondition ? | 194 |
| | Analogie : l'esprit des postconditions – la scène et le rideau | 194 |
| | Principe : jusqu'où faut-il détailler les postconditions ? Analyse agile et analyse lourde | 194 |

| | | |
|------------|--|------------|
| 11.5 | Exemple : postconditions pour saisirArticle | 194 |
| | Création et suppression d'instances | 195 |
| | Modification d'attributs | 195 |
| | Formation et rupture d'associations | 195 |
| 11.6 | Principe : faut-il mettre à jour le Modèle du Domaine ? | 195 |
| 11.7 | Principe : quand les contrats sont-ils utiles ? | 196 |
| 11.8 | Principe : comment créer et rédiger des contrats ? | 196 |
| | Conseils sur la rédaction des contrats | 196 |
| | Quelle est l'erreur la plus fréquente ? | 197 |
| 11.9 | Exemple : contrats du système NextGen | 197 |
| | Opérations système du cas d'utilisation Traiter une vente | 197 |
| | Modifications du modèle du domaine | 198 |
| 11.10 | Exemple : contrats du Monopoly | 198 |
| 11.11 | Application d'UML : opérations, contrats et OCL | 199 |
| | Formulation des contrats d'opérations en OCL | 199 |
| 11.12 | Processus : les contrats d'opération dans le Processus Unifié | 200 |
| 11.13 | Historique | 200 |
| | Prise en charge des contrats par les langages de programmation | 201 |
| 11.14 | Construire le bon système et construire un bon système | 201 |
| 11.15 | Provoquer des changements précoces | 202 |
| 11.16 | Faut-il consacrer des semaines à l'analyse et à la modélisation ? .. | 202 |
| 11.17 | Ressources complémentaires | 202 |
| 12. | Architecture logique et diagrammes de packages UML | 203 |
| 12.1 | Exemple de diagramme de packages | 204 |
| 12.2 | Qu'est-ce qu'une couche ? | 205 |
| 12.3 | De quelles couches traitons-nous dans les études de cas ? | 206 |
| 12.4 | Qu'est-ce que l'architecture logicielle ? | 206 |
| 12.5 | Application d'UML : diagrammes de packages | 207 |
| | Outils UML : rétro-ingénierie, du code aux diagrammes de packages | 208 |
| 12.6 | Principe : concevoir une architecture en couches | 208 |
| | Avantages de la modélisation en couches | 209 |
| | Principe : responsabilités, cohésion et séparation des attributions ... | 210 |

| | |
|---|------------|
| Code : faire correspondre l'organisation du code aux couches et aux packages | 211 |
| Outils UML : rétro-ingénierie, du code aux diagrammes de packages | 211 |
| Définition : couche domaine vs couche application ; objets du domaine | 212 |
| La relation entre la couche domaine et le modèle du domaine ... | 212 |
| Définition : niveaux, couches et partitions | 213 |
| Principe : ne pas représenter les ressources externes comme couche inférieure | 214 |
| 12.7 Principe : le Principe de séparation Modèle-Vue | 214 |
| 12.8 La relation entre les DSS, les opérations système et les couches ... | 216 |
| 12.9 Exemple : architecture logique et diagramme de packages de NextGen | 217 |
| 12.10 Exemple : une architecture logique pour le Monopoly ? | 217 |
| 12.11 Ressources complémentaires | 217 |
| 13. Première étude de la conception objet | 219 |
| 13.1 Modélisation agile et UML allégé | 220 |
| 13.2 Outils de génie logiciel UML | 220 |
| 13.3 Combien de temps consacrer à tracer des diagrammes UML avant de coder ? | 221 |
| 13.4 Conception des objets : modèles statiques et modèles dynamiques | 221 |
| Modélisation dynamique | 222 |
| Modélisation statique | 223 |
| 13.5 Importance des compétences en conception objet par rapport à la notation UML | 223 |
| 13.6 Autres techniques de conception objet : les cartes CRC | 223 |
| 14. Diagrammes d'interaction | 225 |
| 14.1 Diagrammes de séquence et diagrammes de communication | 225 |
| Les points forts et les points faibles des diagrammes de séquence et des diagrammes de communication | 227 |
| Exemple de diagramme de séquence : créer Paiement | 228 |
| 14.2 Exemple de diagramme de communication : créer Paiement | 229 |
| Les modélisateurs novices en UML ne prêtent pas suffisamment d'attention aux diagrammes d'interaction | 229 |

| | | |
|------|---|-----|
| 14.3 | Notation courante des diagrammes d'interaction | 230 |
| | Illustration des participants avec des boîtes de ligne de vie | 230 |
| | Syntaxe de base des messages | 230 |
| | Objets singletons | 231 |
| 14.4 | Notation de base des diagrammes de séquence | 231 |
| | Boîtes de ligne de vie et lignes de vie | 231 |
| | Messages | 231 |
| | Contrôle et barres de spécification d'exécution | 232 |
| | Représentation des réponses ou retours | 232 |
| | Messages d'un objet à lui-même | 233 |
| | Création d'instances | 233 |
| | Lignes de vie et destruction des objets | 234 |
| | Cadres dans les diagrammes de séquence UML | 234 |
| | Boucles | 235 |
| | Messages conditionnels | 235 |
| | Les messages conditionnels en style UML 1.x sont-ils toujours utiles ? | 236 |
| | Messages conditionnels mutuellement exclusifs | 236 |
| | Itérations sur une collection | 236 |
| | Imbrication de cadres | 238 |
| | Mettre en relation les diagrammes d'interaction | 238 |
| | Messages aux classes pour invoquer des méthodes statiques (ou de classe) | 239 |
| | Messages et représentation du polymorphisme | 240 |
| | Appels synchrones et asynchrones | 241 |
| 14.5 | Notation de base des diagrammes de communication | 243 |
| | Liens | 243 |
| | Messages | 243 |
| | Messages d'un objet à lui-même | 244 |
| | Création d'instances | 244 |
| | Affectation de numéros d'ordre aux messages | 245 |
| | Messages conditionnels | 246 |
| | Chemins conditionnels mutuellement exclusifs | 246 |
| | Itérations ou boucles | 247 |
| | Itérations sur une collection | 247 |
| | Messages aux classes pour appeler des méthodes statiques (de classe) | 248 |
| | Messages et représentation du polymorphisme | 248 |
| | Appels synchrones et asynchrones | 249 |

| | |
|---|-----|
| 15. Diagrammes de classes | 251 |
| 15.1 Application d'UML : notation courante des diagrammes de classes | 251 |
| 15.2 Définition : diagrammes de classes de conception (DCC) | 253 |
| 15.3 Définition : classificateurs | 253 |
| 15.4 Représenter les attributs en UML : texte et lignes d'association ... | 254 |
| Quand choisir le texte ou les associations pour représenter les attributs ? | 256 |
| Notation des extrémités d'associations | 256 |
| Représenter les attributs collections : texte et lignes d'associations | 257 |
| 15.5 Symboles d'annotation : notes, commentaires, contraintes et corps de méthodes | 258 |
| 15.6 Opérations et méthodes | 258 |
| Opérations | 258 |
| Représenter les méthodes dans les diagrammes de classes | 259 |
| Problèmes des opérations dans les DCC | 259 |
| 15.7 Mots clés | 260 |
| 15.8 Stéréotypes, profils et étiquettes | 261 |
| 15.9 Propriétés et chaînes de propriétés | 261 |
| 15.10 Généralisation, classes abstraites et opérations abstraites | 262 |
| 15.11 Dépendance | 262 |
| Étiquettes de dépendance | 264 |
| 15.12 Interfaces | 265 |
| 15.13 Composition et agrégation | 265 |
| 15.14 Contraintes | 266 |
| 15.15 Associations qualifiées | 267 |
| 15.16 Classes d'association | 268 |
| 15.17 Classes singletons | 268 |
| 15.18 Classes et interfaces templates | 269 |
| 15.19 Compartiments définis par l'utilisateur | 269 |
| 15.20 Classes actives | 270 |
| 15.21 Quelle est la relation entre diagrammes d'interaction et diagrammes de classes ? | 270 |

| | |
|--|------------|
| 16. GRASP : conception objet et responsabilités | 273 |
| 16.1 UML vs principes de conception | 273 |
| 16.2 Conception objet : entrées, activités et sorties | 273 |
| Les entrées de la conception objet | 274 |
| Les activités de la conception objet | 275 |
| Les sorties | 277 |
| 16.3 Responsabilités et conception pilotée par les responsabilités | 277 |
| 16.4 GRASP : une approche méthodique des bases de la conception OO | 279 |
| 16.5 La relation entre les responsabilités, GRASP et les diagrammes UML | 279 |
| 16.6 Qu'est-ce qu'un pattern ? | 280 |
| Les patterns ont des noms | 281 |
| « Nouveau pattern » est un oxymore | 281 |
| 16.6 L'ouvrage de la « bande des quatre » sur les patterns de conception GRASP est-il un ensemble de patterns ou de principes ? | 281 282 |
| 16.7 En résumé | 282 |
| 16.8 Un court exemple de conception objet avec GRASP | 283 |
| Créateur | 283 |
| Expert en Information | 285 |
| Faible Couplage | 286 |
| Contrôleur | 288 |
| Forte Cohésion | 291 |
| 16.9 Appliquer GRASP à la conception des objets | 292 |
| 16.10 Créateur | 293 |
| 16.11 Expert en Information (ou Expert) | 295 |
| 16.12 Faible Couplage | 300 |
| 16.13 Contrôleur | 304 |
| Interfaces Web et application du pattern Contrôleur côté serveur ... | 309 |
| Implémentation avec Swing : IHM client riche | 311 |
| Implémentation avec Struts : navigateur client et interface Web | 312 |
| 16.14 Forte Cohésion | 316 |
| 16.15 Ressources complémentaires | 321 |
| 17. Conception objet avec les patterns GRASP | 323 |
| La zone de non-magie | 323 |
| 17.1 Qu'est-ce qu'une réalisation de cas d'utilisation ? | 324 |

| | | |
|------------|---|------------|
| 17.2 | Commentaires sur les artefacts | 326 |
| | DSS, opérations système, diagrammes d'interaction et réalisations de cas d'utilisation | 326 |
| | Cas d'utilisation et réalisations des cas d'utilisation | 327 |
| | Contrats d'opération et réalisations de cas d'utilisation | 327 |
| | Modèle du Domaine et réalisations des cas d'utilisation | 328 |
| 17.3 | Réalisations de cas d'utilisation pour l'itération de NextGen en cours | 329 |
| | Initialisation et cas d'utilisation 'Démarrer' | 329 |
| | Concevoir créerNouvelleVente | 329 |
| | Choix de la classe Contrôleur | 330 |
| | Création d'une nouvelle vente | 331 |
| | Conclusion | 331 |
| | Concevoir saisirArticle | 332 |
| | Visibilité par un CatalogueProduits | 334 |
| | La conception finale | 334 |
| | Concevoir terminerVente | 335 |
| | Conception de Vente.getTotal | 338 |
| | Concevoir créerPaiement | 339 |
| | Journalisation des ventes | 340 |
| | Le DCC final pour l'itération 1 de NextGen | 343 |
| | Connecter la couche Présentation à la couche Domaine | 344 |
| | Initialisation et cas d'utilisation Démarrer | 345 |
| 17.4 | Réalisations de cas d'utilisation pour cette itération du Monopoly | 348 |
| | Concevoir lancerPartie | 349 |
| | Choix de la classe contrôleur | 349 |
| | Prendre un tour | 353 |
| | Qui coordonne l'ensemble ? | 354 |
| | Le problème de visibilité | 354 |
| | La conception finale de lancerPartie | 354 |
| | Le principe de Séparation Commande-Requête | 354 |
| | Initialisation et cas d'utilisation de lancement de l'application ... | 357 |
| 17.5 | Processus : conception d'objets itérative et évolutive | 358 |
| | Conception des objets et Processus Unifié | 359 |
| 18. | Concevoir la visibilité et passer de la conception au code | 361 |
| 18.1 | Visibilité entre objets | 362 |
| 18.2 | Qu'est-ce que la visibilité ? | 362 |
| | Visibilité d'attribut | 363 |

| | |
|---|------------|
| Visibilité de paramètre | 364 |
| Visibilité locale | 365 |
| Visibilité globale | 365 |
| 18.3 Programmation et développement itératif et évolutif | 366 |
| Créativité et changements en phase d'implémentation | 366 |
| 18.4 Correspondance entre conception et code | 367 |
| 18.5 Création de définitions de classes à partir des DCC | 367 |
| Définition d'une classe avec des signatures de méthodes et des attributs simples | 367 |
| 18.6 Création des méthodes à partir des diagrammes d'interaction ... | 368 |
| 18.7 Classes collection dans le code | 369 |
| 18.8 Traitement des exceptions et des erreurs | 370 |
| 18.9 Définition de la méthode Vente.créerLigneArticles | 370 |
| 18.10 Ordre de l'implémentation | 371 |
| 18.11 Développement piloté par les tests ou avec tests préalables | 372 |
| 18.12 En résumé | 372 |
| 18.13 Présentation de la solution programmatique de NextGen | 372 |
| La classe Paiement | 372 |
| La classe CatalogueProduits | 373 |
| La classe Registre | 373 |
| La classe DescriptionProduit | 374 |
| La classe Vente | 374 |
| La classe LigneArticles | 375 |
| La classe Magasin | 375 |
| 18.14 Présentation de la solution programmatique du Monopoly | 376 |
| La classe Case | 376 |
| La classe Pion | 377 |
| La classe Dé | 377 |
| La classe Plateau | 377 |
| La classe Joueur | 379 |
| La classe JeuDeMonopoly | 379 |
| 19. Développement piloté par les tests et refactorisation | 381 |
| 19.1 Développement piloté par les tests | 381 |
| Prise en charge du développement piloté par les tests et de xUnit par les IDE | 384 |

| | | |
|---|--|------------|
| 19.2 | Refactorisation | 385 |
| | Exemple | 386 |
| | Prise en charge de la refactorisation par les IDE | 388 |
| 19.3 | Ressources complémentaires | 389 |
| 20. | Outils UML et UML en mode plan | 391 |
| 20.1 | Pro-ingénierie, rétro-ingénierie et ingénierie cyclique | 392 |
| 20.2 | Les fonctions intéressantes | 392 |
| 20.3 | Que doit-on rechercher dans un outil ? | 393 |
| 20.4 | Avec UML en mode esquisse, comment actualiser les diagrammes après le codage ? | 393 |
| 20.5 | Ressources complémentaires | 394 |
| Partie IV Élaboration de l'itération 2 | | |
| 21. | Itération 2 – Rapide mise à jour des exigences et de l'analyse .. | 397 |
| 21.1 | De l'itération 1 à l'itération 2 | 398 |
| | Simplifications de l'étude de cas | 399 |
| 21.2 | Étude de cas : NextGen | 399 |
| | Cas d'utilisation | 399 |
| | Diagrammes de séquence système (DSS) | 399 |
| | Modèle du Domaine | 400 |
| | Contrats d'opérations système | 401 |
| 21.3 | Étude de cas : Monopoly | 401 |
| | Cas d'utilisation, etc. | 401 |
| | Modèle du Domaine | 401 |
| 21.4 | Exigences de l'itération 2 – Conception objet et patterns | 403 |
| | NextGen | 403 |
| | Développement incrémental des cas d'utilisation au fur et à mesure des itérations | 404 |
| | Monopoly | 404 |
| 22. | GRASP : autres patterns d'affectation des responsabilités | 407 |
| 22.1 | Polymorphisme | 408 |
| 22.2 | Fabrication Pure | 414 |
| 22.3 | Indirection | 419 |
| 22.4 | Protection des variations (PV) | 420 |

| | |
|---|-----|
| 23. Application des patterns GoF | 429 |
| 23.1 Adaptateur (GoF) | 429 |
| 23.2 Certains principes GRASP sont des généralisations d'autres patterns | 431 |
| 23.3 Découvertes « d'analyse » lors de la conception : Modèle du Domaine | 433 |
| 23.4 Fabrique Concrète | 433 |
| 23.5 Singleton (GoF) | 435 |
| Problèmes d'implémentation et de conception | 437 |
| 23.6 Conclusion sur les services externes dotés d'interfaces différentes . | 439 |
| 23.7 Stratégie (GoF) | 440 |
| Création d'une Stratégie avec une Fabrique Concrète | 441 |
| En résumé | 443 |
| 23.8 Le pattern GoF Composite et autres principes de conception | 444 |
| Création de StratégieTarification multiples | 448 |
| En résumé | 451 |
| 23.9 Façade (GoF) | 452 |
| En résumé | 454 |
| 23.10 Observateur/Diffusion-Souscription/Modèle de délégation d'événements (GoF) | 455 |
| Pourquoi parle-t-on d'Observateur, de Diffusion-Souscription ou de Délégation d'événements ? | 458 |
| L'Observateur ne sert pas uniquement à connecter les objets de l'IHM et du modèle | 459 |
| Un même diffuseur peut avoir plusieurs souscripteurs pour un événement | 460 |
| Implémentation | 460 |
| En résumé | 461 |
| 23.11 Ressources complémentaires | 462 |

Partie V

Élaboration de l'itération 3

| | |
|--|-----|
| 24. Itération 3 – Compléments sur les cas d'utilisation | 465 |
| 24.1 Système POS NextGen | 465 |
| 24.2 Monopoly | 466 |

| | | |
|------------|---|------------|
| 24.3 | Relations entre cas d'utilisation | 466 |
| | Principe : évitez de consacrer trop de temps aux relations entre cas d'utilisation | 467 |
| 24.4 | La relation include | 467 |
| | Utilisation d'include avec le traitement des événements asynchrones | 469 |
| | En résumé | 470 |
| 24.5 | Terminologie : cas d'utilisation concrets, abstraits, de base et additionnels | 470 |
| 24.6 | La relation extend | 470 |
| 24.7 | La relation de généralisation | 472 |
| 24.8 | Diagrammes de cas d'utilisation | 472 |
| 24.9 | Mise à jour de NextGen | 473 |
| | Nouveaux diagrammes de séquence système | 473 |
| | Nouvelles opérations système | 475 |
| | Nouveaux contrats d'opérations système | 476 |
| 25. | Diagrammes d'activités et diagrammes d'états UML | 477 |
| 25.1 | Exemple de diagramme d'activités | 477 |
| 25.2 | Appliquer les diagrammes d'activités | 478 |
| | Modélisation des processus métier | 478 |
| | Modélisation du flot de données | 479 |
| | Programmation concurrente et modélisation d'algorithmes parallèles | 480 |
| 25.3 | Compléments sur la notation des diagrammes d'activités UML .. | 481 |
| 25.4 | Principes | 482 |
| 25.5 | Exemple : diagramme d'activités de NextGen | 482 |
| 25.6 | Processus : diagrammes d'activités dans le Processus Unifié | 483 |
| 25.7 | Historique des diagrammes d'activités | 484 |
| 25.8 | Exemple de machines à états | 484 |
| 25.9 | Définitions : événements, états et transitions | 485 |
| 25.10 | Appliquer des diagrammes d'états | 485 |
| | Objets état-dépendants et état-indépendants | 485 |
| | Modélisation d'objets état-dépendants | 486 |

| | | |
|------------|--|------------|
| 25.11 | Informations supplémentaires sur la notation des machines à états UML | 487 |
| | Actions de transition et gardes | 487 |
| | États imbriqués | 488 |
| 25.12 | Exemple : modélisation de la navigation dans une interface utilisateur avec des machines à états | 489 |
| 25.13 | Exemple : machines à états d'un cas d'utilisation de NextGen | 490 |
| 25.14 | Processus : diagrammes d'états dans le Processus Unifié | 490 |
| 25.15 | Ressources complémentaires | 490 |
| 26. | Affinement du Modèle du Domaine | 491 |
| 26.1 | Nouveaux concepts du Modèle du Domaine de NextGen | 491 |
| | Listes de catégories de concepts | 492 |
| | Identification des groupes nominaux à partir des cas d'utilisation | 492 |
| | Transactions avec les services d'autorisation | 493 |
| 26.2 | Généralisation | 493 |
| 26.3 | Définitions des super-classes et des sous-classes conceptuelles | 495 |
| | Généralisation et définition des classes conceptuelles | 495 |
| | Généralisation et ensembles de classes | 495 |
| | Conformité des définitions des sous-classes conceptuelles | 496 |
| | Conformité des sous-classes conceptuelles aux règles des ensembles . | 497 |
| | Qu'est-ce qu'une sous-classe conceptuelle correcte ? | 497 |
| 26.4 | Quand définir une sous-classe conceptuelle ? | 497 |
| | Raisons de partitionner une classe conceptuelle en sous-classes . | 498 |
| 26.5 | Quand définir une super-classe conceptuelle ? | 499 |
| 26.6 | La hiérarchie des classes conceptuelles du système POS NextGen ... | 500 |
| | Classes de paiement | 500 |
| | Classes de services d'autorisation | 500 |
| | Transactions avec les services d'autorisation | 500 |
| 26.7 | Classes conceptuelles abstraites | 502 |
| | Notation des classes abstraites en UML | 503 |
| 26.8 | Modélisation des changements d'état | 504 |
| 26.9 | Hiéarchies de classes et héritage au niveau de l'implémentation . | 504 |
| 26.10 | Classes d'associations | 505 |
| 26.11 | Agrégation et composition | 508 |
| | Identifier la composition | 508 |

| | |
|---|------------|
| Avantage de la représentation de la composition | 509 |
| Composition dans le Modèle du Domaine de NextGen | 509 |
| 26.12 Intervalles de temps et prix des produits – Correction d'une « erreur » de l'itération 1 | 510 |
| 26.13 Noms de rôles | 510 |
| 26.14 Rôles conceptuels et rôles dans les associations | 511 |
| 26.15 Éléments dérivés | 512 |
| 26.16 Associations qualifiées | 513 |
| 26.17 Associations réflexives | 514 |
| 26.18 Utilisation des packages pour organiser le Modèle du Domaine .. | 514 |
| Partitionner le Modèle du Domaine | 515 |
| Packages du Modèle du Domaine du POS | 516 |
| Package Communs/Divers | 516 |
| Paiements | 517 |
| Produits | 518 |
| Ventes | 518 |
| Transactions d'autorisation | 519 |
| 26.19 Exemple : affinements du Modèle du Domaine du Monopoly | 520 |
| 27. Analyse architecturale | 521 |
| 27.1 Processus : quand commencer l'analyse architecturale ? | 522 |
| 27.2 Définition : points de variation et points d'évolution | 522 |
| 27.3 L'analyse architecturale | 522 |
| 27.4 Étapes de base de l'analyse architecturale | 523 |
| 27.5 La science : identification et analyse des facteurs architecturaux .. | 524 |
| Facteurs architecturaux | 524 |
| Scénarios de qualité | 525 |
| Description des facteurs | 526 |
| Facteurs et artefacts du Processus Unifié | 526 |
| 27.6 Exemple partiel de tableau des facteurs architecturaux pour NextGen | 528 |
| 27.7 L'art : solutions aux facteurs architecturaux | 528 |
| Traçabilité des alternatives, décisions et motivations | 528 |
| Priorités | 532 |
| Principes fondamentaux de la conception architecturale | 534 |
| Patterns architecturaux | 537 |

| | | |
|------------|---|------------|
| 27.8 | Résumé des principaux thèmes de l'analyse architecturale | 537 |
| 27.9 | Processus : analyse architecturale et Processus Unifié | 538 |
| | Informations sur l'architecture dans les artefacts du Processus Unifié | 538 |
| | Phases | 539 |
| 27.10 | Ressources complémentaires | 539 |
| 28. | Affinement de l'architecture logique | 541 |
| 28.1 | Exemple : l'architecture logique de NextGen | 541 |
| | Couplage entre couches et entre packages | 543 |
| | Scénarios d'interactions entre couches et entre packages | 545 |
| 28.2 | Collaborations avec le pattern Couches | 546 |
| | Packages simples et sous-systèmes | 546 |
| | Façade | 547 |
| | Façades de session et couche Application | 548 |
| | Contrôleur | 549 |
| | Opérations système et couches | 549 |
| | Collaboration ascendante avec le pattern Observateur | 550 |
| | Couplage lâche des couches | 551 |
| 28.3 | Autres problèmes du pattern Couches | 552 |
| | Vues logiques, vues des processus et vues de déploiement | 553 |
| | La couche Application est-elle optionnelle ? | 553 |
| | Appartenance à des ensembles flous dans différentes couches | 554 |
| | Contre-indications et obligations des couches | 554 |
| | Usages connus | 555 |
| | Patterns connexes | 557 |
| | Alias | 557 |
| 28.4 | Séparation Modèle-Vue et communication « ascendante » | 557 |
| 28.5 | Ressources complémentaires | 558 |
| 29. | Conception des packages | 559 |
| 29.1 | Principes d'organisation des packages | 560 |
| | Principe : organiser les packages en partitions verticales et horizontales fonctionnellement cohésives | 560 |
| | Principe : packager une famille d'interfaces | 561 |
| | Principe : créer un package par tâche et par groupe de classes instables | 561 |
| | Principe : les plus responsables sont les plus stables | 562 |

| | |
|--|------------|
| Principe : factoriser les types indépendants | 563 |
| Principe : utiliser des Fabrications pour limiter la dépendance aux packages concrets | 563 |
| Principe : pas de cycles dans les packages | 564 |
| 29.2 Ressources complémentaires | 565 |
| 30. Application des patterns GoF et GRASP à la conception des objets | 567 |
| 30.1 Exemple : NextGen | 567 |
| 30.2 Basculement sur les services locaux, mise en cache locale et performances | 568 |
| 30.3 Gestion des erreurs | 573 |
| Lancement des exceptions | 574 |
| Exceptions en UML | 575 |
| Traitement des erreurs | 576 |
| 30.4 Basculement sur les services locaux avec une Procuration (Proxy) [GoF] | 579 |
| 30.5 Conception des besoins non fonctionnels ou de qualité | 581 |
| 30.6 Accès aux équipements physiques externes via des adaptateurs ... | 582 |
| 30.7 Fabrique Abstraite (GoF) pour les familles d'objets apparentés ... | 584 |
| Une classe abstraite Fabrique Abstraite | 585 |
| 30.8 Gestion des paiements avec les patterns Polymorphisme et Faire soi-même | 587 |
| Classes à granularité fine | 589 |
| Autorisations de paiement à crédit | 589 |
| 30.9 Exemple : Monopoly | 593 |
| 30.10 Conclusion | 596 |
| Attention à la « patternite » | 596 |
| 31. Diagrammes de déploiement et diagrammes de composants .. | 597 |
| 31.1 Diagrammes de déploiement | 597 |
| 31.2 Diagrammes de composants | 599 |
| 32. Documentation de l'architecture : UML et le modèle N+1 vues .. | 601 |
| 32.1 Le Document d'Architecture du Logiciel et ses vues architecturales | 601 |
| Document d'Architecture du Logiciel | 601 |

| | |
|---|------------|
| Motivation : pourquoi créer un Document d'Architecture du Logiciel ? | 602 |
| Vues architecturales | 602 |
| Le modèle N+1 (ou 4+1) vues | 603 |
| Vues architecturales détaillées | 603 |
| 32.2 Notation : structure du Document d'Architecture du Logiciel | 605 |
| 32.3 Exemple : Document d'Architecture du Logiciel du système POS NextGen | 606 |
| 32.4 Exemple : Document d'Architecture du Logiciel de Struts | 611 |
| 32.5 Processus : documentation architecturale itérative | 615 |
| Processus Unifié et Document d'Architecture du Logiciel | 615 |
| 32.6 Ressources complémentaires | 616 |
| 33. Développement itératif et gestion de projet agile | 617 |
| 33.1 Planifier une itération | 617 |
| 33.2 Planification adaptative et planification prédictive | 618 |
| 33.3 Plans de phases et plans d'itération | 620 |
| 33.4 Planifier des itérations avec des cas d'utilisation et des scénarios | 621 |
| 33.5 Validité et invalidité des estimations précoces | 623 |
| 33.6 Organisation des artefacts | 624 |
| 33.7 Ressources complémentaires | 625 |
| Annexes | |
| Glossaire | 633 |
| Bibliographie | 639 |
| Index | 647 |