

Concevoir une application d'envergure

1. Le travail de conception : un gain de temps considérable	13
1.1 Un exemple simple	13
1.2 Diviser pour régner	18
1.3 Exemple d'amélioration	21
1.4 Approche Modèle-Vue-Contrôleur (MVC)	26
2. Démarrage	29
2.1 Choix du jeu	29
2.2 Environnement de développement	33
2.2.1 NetBeans avec EasyUML	33
2.2.2 Autres environnements	39
2.3 Projets exemple	40

Représenter l'état du jeu

1. Les données et leur représentation	41
1.1 Conception initiale	41
1.2 Types d'informations	42
1.3 Exemple avec le jeu Pacman	42

1.3.1 Éléments fixes	43
1.3.2 Éléments mobiles	43
1.4 Développement du jeu vidéo : cahier des charges	44
2. Les informations élémentaires	45
2.1 Classes	45
2.1.1 Créer un nouveau diagramme de classes avec NetBeans/EasyUML	46
2.1.2 Ajout d'une classe	50
2.1.3 Ajout d'une énumération	51
2.2 Hiérarchie de classes (polymorphisme)	53
2.2.1 Classe mère	54
2.2.2 Éléments statiques	57
2.2.3 Accesseurs et mutateurs (getters et setters)	59
2.3 Combinaison de propriétés (Composition)	60
2.3.1 Interfaces	60
2.3.2 Composition	63
2.3.3 Interfaces et Composition	64
2.4 Exercices	66
2.4.1 Exercice : Jeu de rôle	66
2.4.2 Exercice : Jeu de rôle avec classes multiples	66
2.5 Développement du jeu vidéo : classes d'éléments	67

3. Les conteneurs	67
3.1 Les listes et tableaux associatifs	67
3.1.1 Listes	68
3.1.2 Piles et files	70
3.1.3 Tableaux associatifs	72
3.2 Contenir des éléments épars (Patron Décorateur)	74
3.3 Graphes d'éléments	79
3.4 Les tableaux multidimensionnels	81
3.5 Parcourir les conteneurs (Patron Itérateur)	84
3.6 Conteneur principal	89
3.7 Exercices	91
3.7.1 Exercice : Civilization	91
3.7.2 Exercice : Stellaris	92
3.8 Développement du jeu vidéo : les conteneurs	92
4. Tests unitaires	93
4.1 Implanter des tests unitaires	93
4.1.1 Génération des accesseurs/mutateurs (getters/setters)	93
4.1.2 Générer le code source Java	95
4.1.3 Écriture des tests unitaires	96
4.2 Exercices	

4.2.1 Exercice : Test de Civilization	102
4.2.2 Exercice : Test de Stellaris	102
4.3 Développement du jeu vidéo : tests unitaires état	105
5. Solutions exercices	106
5.1 Exercice 2.4.1 : Jeu de rôle	106
5.2 Exercice 2.4.2 : Jeu de rôle avec classes multiples	107
5.3 Exercice 3.7.1 : Civilization	108
5.4 Exercice 3.7.2 : Stellaris	110
5.5 Exercice 4.2.1 : Test de Civilization	111
5.6 Exercice 4.2.2 : Test de Stellaris	112

Interface utilisateur

1. Interface utilisateur 2D avec AWT	115
1.1 Affichage synchrone en double tampon	116
1.1.1 Afficher une fenêtre	116
1.1.2 Création d'un canevas	118
1.1.3 Double tampon et boucle principale (Patron Boucle de Jeu)	119
1.1.4 Synchronisation	123
1.2 Affichage avec des tuiles	127

1.2.1 Fabriquer un monde en tuiles	128
1.2.2 Charger et afficher une image	132
1.2.3 Dessiner avec des tuiles	134
1.2.4 Dessiner un monde en tuiles	136
1.2.5 Animations avec des tuiles	140
1.3 Contrôles	142
1.3.1 Contrôles AWT (Patron Observateur)	142
1.3.2 Clavier	144
1.3.3 Souris	147
1.4 Optimisation de l'affichage	151
1.4.1 Performance de l'affichage	151
1.4.2 Utiliser un lanceur	154
2. Interface utilisateur 3D avec LWJGL	157
2.1 Création d'une fenêtre	158
2.1.1 Initialisation	158
2.1.2 Exécution et destruction	160
2.2 Afficher un triangle	162
2.2.1 Pipeline d'affichage	162
2.2.2 Pipeline pour afficher un Triangle	163
2.2.3 Gérer les Shaders	169

2.3 Coloriser et indexer	172
2.3.1 Coloriser	172
2.3.2 Indexer	175
2.3.3 Exemple avec le dessin d'un disque	177
2.4 Perspective et contrôles	178
2.4.1 Perspective	178
2.4.2 Caméra	181
2.4.3 Contrôles	183
2.5 Transformation et animations	187
2.5.1 Transformer un objet	187
2.5.2 Animation	188
2.6 Textures	189
2.6.1 Charger une texture	189
2.6.2 Utiliser la texture	190
3. Conception logicielle	192
3.1 Abstraction de l'interface utilisateur	192
3.1.1 Patron Façade	193
3.1.2 Calques d'affichage (Patron Fabrique)	198
3.1.3 Contrôles	206
3.2 Modes de jeu	210

3.2.1 Abstraire la notion de mode de jeu (Patron Template)	211
3.2.2 Changer de mode de jeu (Patron État)	215
3.2.3 Proposer un menu	222
3.2.4 Cacher les polices (Patron Poids-Mouche)	226
3.2.5 Faire une suite de menus (Patron État)	229
3.3 Exercices	232
3.3.1 Exercice 3.3.1 : Afficher du texte avec la façade	232
3.3.2 Exercice 3.3.2 : Afficher une image avec la façade	233
3.3.3 Exercice 3.3.3 : Détecter les séquences de touches	234
4. Développement du jeu vidéo : interface utilisateur	236
5. Solutions des exercices	237
5.1 Exercice 3.3.1 : Afficher du texte avec la façade	237
5.2 Exercice 3.3.2 : Afficher une image avec la façade	239
5.3 Exercice 3.3.3 : Détecter les séquences de touches	241
Moteur de jeu	
1. Approche générale	245
1.1 Présentation	245
1.2 Motivation	246

2. Synchronisation entre état et interface utilisateur	248
2.1 Dessiner l'état du jeu	252
2.1.1 Initialiser le monde (Patron Fabrique Abstraite)	252
2.1.2 Définir le jeu de tuiles (Patron Poids-Mouche)	257
2.1.3 Dessiner le monde contenu dans l'état du jeu	261
2.2 Déplacement et animations	264
2.2.1 Déplacement des personnages	264
2.2.2 Animations (Patron Composite)	272
2.3 Séparer état et rendu	276
2.3.1 Rendre les animations indépendantes (Patron Observateur)	276
2.3.2 Interpoler les déplacements	283
2.4 Exercices	288
2.4.1 Exercice 2.4.1 : Fabriquer une galaxie	288
2.4.2 Exercice 2.4.2 : Déplacer une armée	289
2.5 Développement du jeu vidéo : dessiner l'état	289
3. Règles du jeu	291
3.1 Définir les règles du jeu	291
3.2 Exemple de définition de règles avec le jeu Pacman	292
3.2.1 Horloge globale	292
3.2.2 Changements extérieurs	

3.2.3 Changements autonomes	293
3.2.4 Fin de partie	293
3.3 Appliquer les règles (Patron Commande)	295
3.3.1 Patron Commande	295
3.3.2 Initialiser le jeu avec une commande	297
3.3.3 Déplacer les personnages avec des commandes	300
3.3.4 Implanter toutes les règles	306
3.4 Faire et défaire	309
3.4.1 Défaire avec le patron Memento	309
3.4.2 Défaire par compensation	315
3.5 Tester le moteur de règles	324
3.5.1 Copier en profondeur (Patron Prototype)	324
3.5.2 Comparer en profondeur	327
3.5.3 Implanter les tests	329
3.6 Exercices	333
3.6.1 Exercice 3.6.1 : Jouer à plusieurs sur le même clavier	333
3.6.2 Exercice 3.6.2 : Jeu de dames avec le patron Commande	333
3.6.3 Développement du jeu vidéo : règles du jeu	333
4. Fonctionnalités supplémentaires	335
4.1 Paramétrer une partie (Patron Constructeur)	

4.2 Charger un niveau (Patron Visiteur)	335
4.3 Charger/sauvegarder l'état du jeu (Patron Proxy)	339
	345

5. Solutions des exercices

	354
5.1 Exercice 2.4.1 : Fabriquer une galaxie	354
5.2 Exercice 2.4.2 : Déplacer une armée	358
5.3 Exercice 3.6.1 : Jouer à plusieurs sur le même clavier	360
5.4 Exercice 3.6.2 : Jeu de dames avec le patron Commande	361

Intelligence Artificielle

1. Préparation

	363
1.1 État du jeu non modifiable	363
1.1.1 Approche avec le Patron Proxy	364
1.1.2 Approche avec le Patron Décorateur	366
1.2 Interface Intelligence Artificielle	370
1.2.1 Dresser la liste des commandes possibles (Patron Stratégie)	370
1.2.2 Comportement aléatoire	372
1.2.3 Évaluation	374
1.3 Exercices	375
1.3.1 Exercice 1.3.1 : Galaxie non modifiable	375

1.3.2 Exercice 1.3.2 : Puissance 4 avec une IA simple	375
1.4 Développement du jeu vidéo : préparation IA	376
2. Intelligence Artificielle sans planification	377
2.1 Heuristique simple	377
2.2 Cartes de distance	379
2.2.1 Rappels sur les graphes	381
2.2.2 Calcul des cartes de distance	384
2.2.3 Implantation	389
2.3 Gestion du calcul des cartes de distance	392
2.3.1 Suivre un personnage	392
2.3.2 Fournisseur de services	395
2.4 IA par comportement	400
2.4.1 Principe	400
2.4.2 Exemple avec Pacman	401
2.5 Exercices	404
2.5.1 Exercice 2.5.1 : Plus court chemin dans la galaxie	404
2.5.2 Exercice 2.5.2 : Puissance 4 avec une IA heuristique	405
2.6 Développement du jeu vidéo : IA sans planification	405
3. Intelligence Artificielle avec planification	406
3.1 Parcourir les états futurs	

	406
3.1.1 Graphes d'états	406
3.1.2 Les arbres de recherche	408
3.1.3 Algorithmes de parcours	411
3.2 Minimax	414
3.2.1 Version exhaustive	414
3.2.2 Implantation	417
3.2.3 Optimisation Alpha Bêta	423
3.2.4 Validation avec le jeu du Morpion	427
3.2.5 Application pour Pacman	431
3.3 Autres approches classiques en Intelligence artificielle	437
3.4 Exercices	439
3.4.1 Exercice 3.4.1 : Parcours avec l'interface Node	439
3.4.2 Exercice 3.4.2 : Puissance 4 avec une IA avec planification	439
3.5 Développement du jeu vidéo : IA avec planification	440
4. Solution des exercices	441
4.1 Exercice 1.3.1 : Galaxie non modifiable	441
4.2 Exercice 1.3.2 : Puissance 4 avec une IA simple	444
4.3 Exercice 2.5.1 : Plus court chemin dans la galaxie	446
4.4 Exercice 2.5.2 : Puissance 4 avec une IA heuristique	448
4.5 Exercice 3.4.1 : Parcours avec l'interface Node	

4.6 Exercice 3.4.2 : Puissance 4 avec une IA avec planification	450
	453

Exécution concurrente et réseaux

1. Exécution concurrente

1.1 Séparer Moteur de règles et Interface utilisateur	457
1.1.1 Échanges entre les acteurs	457
1.1.2 Thread du moteur de règles	458
1.1.3 Transférer les commandes (Double Tampon)	462
1.1.4 Mise à disposition du moteur et de l'état (Patron Observateur)	464
1.1.5 Cacher les notifications	467
1.2 Paralléliser les traitements	470
1.2.1 Appels asynchrones (Patrons Observateur et Futur)	471
1.2.2 Patron Pool de threads	474
1.2.3 Patron Producteur/Consommateur	479
1.2.4 Parallélisation Minimax	486
1.3 Exercices	492
1.3.1 Exercice 1.3.1 : Paralléliser la recherche exhaustive de collisions	492
1.3.2 Exercice 1.3.2 : Paralléliser la recherche indexée de collisions	494
1.4 Développement du jeu vidéo : exécution concurrente	495

2. Communication réseau	498
2.1 Notions essentielles	499
2.1.1 Couches réseau	499
2.1.2 Ethernet	500
2.1.3 IP (Internet Protocol)	501
2.1.4 UDP (User Datagram Protocol)	505
2.1.5 TCP (Transmission Control Protocol)	506
2.1.6 HTTP (Hypertext Transfer Protocol)	507
2.1.7 API Web REST (Representational State Transfer)	513
2.1.8 Formats de données	517
2.2 Implanter des services web	521
2.2.1 Créer un serveur HTTP	521
2.2.2 En-têtes HTTP et format des données	523
2.2.3 Produire du JSON (Patrons Fabrique et Constructeur)	524
2.2.4 Méthode HTTP et Client	526
2.2.5 Service GET	528
2.2.6 Service POST	534
2.3 Exercices	537
2.3.1 Exercice 2.3.1 : Créer un chat avec des sockets	537
2.3.2 Exercice 2.3.2 : Implanter un mini serveur HTTP	538

3. Jeu en réseau	540
3.1 Principes	540
3.2 Sérialiser les commandes	542
3.2.1 Sérialisation	542
3.2.2 Désérialisation	545
3.3 Rassembler les joueurs	547
3.3.1 Définition du service	547
3.3.2 Implantation des services	548
3.3.3 Gestionnaire de services	551
3.3.4 Serveur de services	553
3.3.5 Requêtes depuis les clients	556
3.3.6 Menus graphiques et requêtes réseau	558
3.4 Jeu multijoueur	562
3.4.1 Implantation côté serveur	562
3.4.2 Implantation côté client	568
3.4.3 Améliorations	572
3.5 Exercices	574
3.5.1 Exercice 3.5.1 : Multijoueur avec UDP	574
3.5.2 Exercice 3.5.2 : Multijoueur sans état local	576
3.6 Développement du jeu vidéo : jeu en réseau	579

4. Solutions des exercices	583
4.1 Exercice 1.3.1 : Paralléliser la recherche exhaustive de collisions	583
4.2 Exercice 1.3.2 : Paralléliser la recherche indexée de collisions	585
4.3 Exercice 2.3.1 : Créer un chat avec des sockets	588
4.4 Exercice 2.3.2 : Implanter un mini serveur HTTP	590
4.5 Exercice 3.5.1 : Multijoueur avec UDP	600
4.6 Exercice 3.5.2 : Multijoueur sans état local	606
Index	611