INSA de Rennes
Quatrième année - Informatique

Projet d'Analyse, Conception et POO

Rapport de conception

Mickaël Olivier, Benoit Travers

18 novembre 2013

Table des matières

1 Cas d'utilisation **4**
1.1 Création d'une partie 4
1.2 Tour de jeu 5

2 Diagrammes de classe **7**
2.1 Modélisation globale du jeu 7
2.2 Patrons utilisés 7

3 Diagramme d'états-transitions **11**

4 Diagrammes d'intéraction **12**
4.1 Création de partie 12
4.2 Déplacer ou attaquer une unité 13

Appendices **14**

Annexe A Diagramme de classes **15**

Annexe B Diagramme de séquence : Création de partie **16**

Annexe C Diagramme de séquence : Déplacer ou attaquer une unité **17**

Introduction

En cette quatrième année Informatique, nous sommes amenés à modéliser un jeu tour-par-tour similaire à Small World.L'une des premières étape dans la réalisation d'un tel projet est la modélisation à l'aide des différents types de diagramme UML. Ce rapport marque la fin de cette phase d'analyse et de conception, regroupant les différents diagrammes de cas d'utilisation, diagrammes de classe, diagramme d'états-transitions et diagrammes d'intéraction.

Un rapide rappel du principe du jeu à implémenter s'impose. Deux joueurs, que l'on appelera joueur A et joueur B, gère un peuple (Nains, Gaulois et Vikings) composé de plusieurs unités. L'objectif est de gérer ses unités sur une carte, que se partage les deux joueurs, pour obtenir le plus de points possible à la fin d'un certain nombre de tours. Initialement, les joueurs positionnent leurs unités sur une case, un placement qui rapportera plus ou moins de points. Ensuite, les joueurs devront déplacer leurs unités, encore une fois l'occupation d'une case rapporte plus ou moins de points, et attaquer les unités de leur adversaire dans l'objectif de limiter l'acquisition de points de l'adversaire. Après un certain nombre de tours la partie s'arrête et les points sont comptés.

FIGURE 1 – Plateau du jeu Small World

1 Cas d'utilisation

Premièrement, nous retrouvons un diagramme de cas d'utilisation illustrant la globalité du jeu (Figure 2). En se penchant sur ce diagramme, on remarque qu'un joueur peut créer une partie ou il peut jouer. Les diagrammes de cas d'utilisation pour la création de la partie et le tour de jeu sont décrit par la suite.

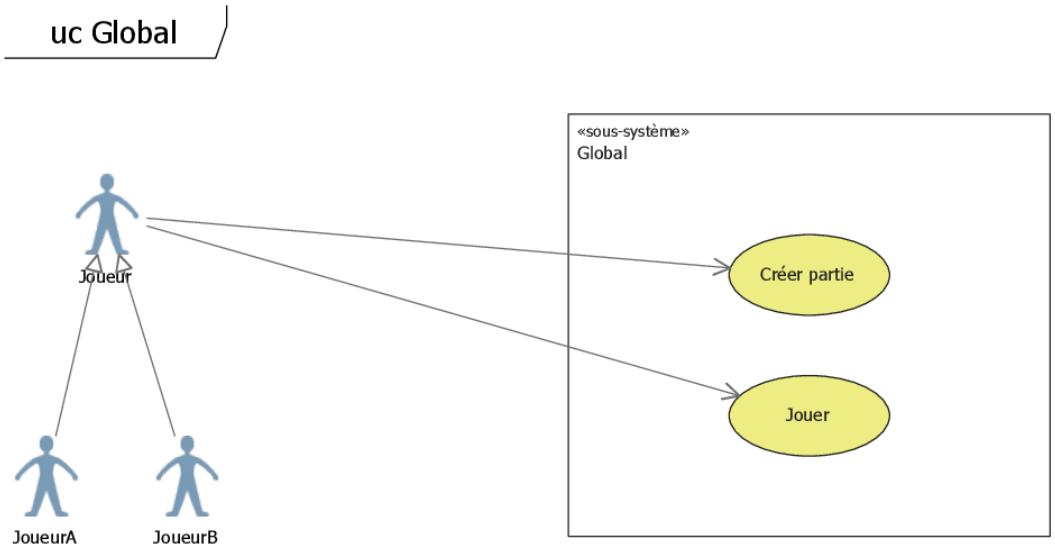


FIGURE 2 – Diagramme de cas d'utilisation global

1.1 Création d'une partie

Avant de pouvoir jouer, il faut créer la partie. Pour cela, le joueur A choisit la carte, à savoir, soit une carte de démonstration, soit une petite carte, soit une carte de taille normale. Ensuite, les joueurs devront choisir leur peuple parmis les Vikings, les Gaulois et les Nains. Cependant, les joueurs ne peuvent pas sélectionner le même peuple. Une fois la carte et les peuples choisis, il suffit de lancer la partie puis de positionner les unités sur une case de la carte. Le positionnement des unités peut être impossible selon la case choisie. Toutes ces actions sont illustrées par le diagramme de cas d'utilisation de la Figure 3.

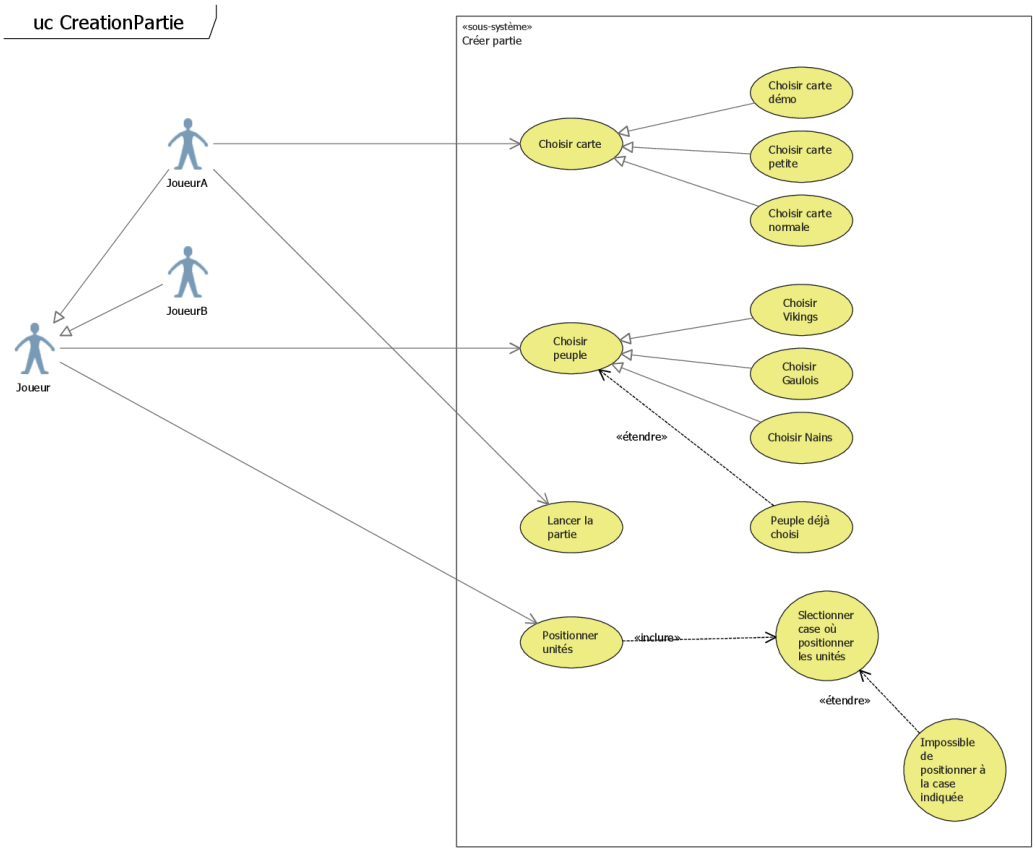


FIGURE 3 – Diagramme de cas d'utilisation : Création de la partie

1.2 Tour de jeu

Lors de son tour de jeu, un joueur peut effectuer différentes actions. Il peut déplacer une unité qui n'a pas passé son tour et pour se faire, il doit sélectionner la case contenant l'unité, il chosit l'unité sur la case selectionnée (il peut y avoir plusieurs unités d'un même joueur sur une même case) puis il sélectionne une case de destination. Cependant, il est necessaire de veiller à ce que l'unité sélectionnée par le joueur puisse effectuer le déplacement (l'unité à les points de mouvement necessaires et la type de la case de destination est compatible avec le peuple du joueur).

Un joueur peut aussi attaquer une unité adverse avec une de ses unités qui n'a pas passé son tour. Pour cela, le joueur doit séléctionner la case contenant l'unité qu'il souhaite engager au combat, il choisit l'unité sur la case selectionnée puis il selectionne la case de destination avec l'unité adversaire qu'il souhaite attaquer. Encore une fois, il faut veiller à ce que l'unité sélectionnée par le joueur puisse attaquer l'adversaire.

Enfin, un joueur peut passer le tour d'une unité n'ayant pas déjà passé son tour en selectionnant la case contenant l'unité puis en choisissant l'unité sur cette case. Mais un joueur peut simplement passer son tour sans avoir besoin de passer le tour de toutes ses unités. Ces actions sont illustrées dans le diagramme de cas d'utilisation de la Figure 4.

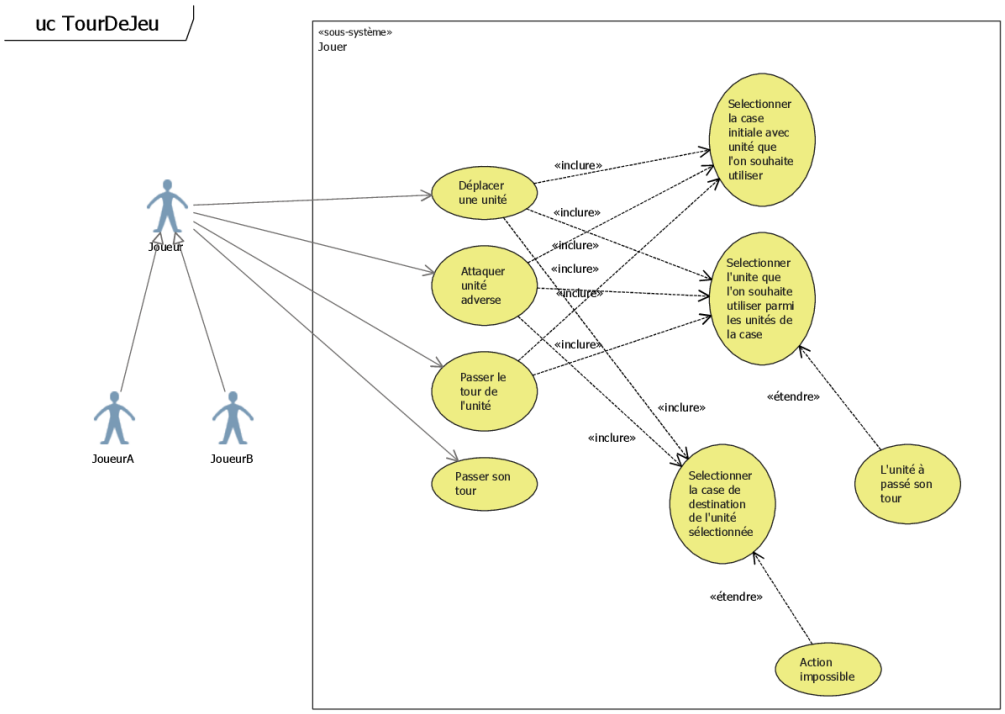


FIGURE 4 – Diagramme de cas d'utilisation : Tour de Jeu

2 Diagrammes de classe

2.1 Modélisation globale du jeu

En annexe A, on retrouve le diagramme de classe de notre jeu avec les différents patrons utilisés. Par la suite, nous allons détailler les différents patrons utilisés.

2.2 Patrons utilisés

Le premier patron utilisé est la fabrique. La classe Peuple est considérée comme une fabrique d'Unité. Deux des classes héritées de Peuple seront instanciées au moment de la création des unités des joueurs dans le MonteurPartie puis ces classes Peuple ne seront pas réutilisées par la suite.

En ce qui concerne la classe Unite, on remarque qu'elle contient de nombreux attributs tels que ses coordonnées ("_x" et "_y"), ses caractéristiques ("_attaque", "_défense", "_pdv" pour points de vie et "_pm" pour points de mouvement) et un attribut indiquant si l'unité a passé sont tour ("_passeTour"). Cette classe Unite contient aussi toutes les méthodes pour se positionner en début de partie, se déplacer et attaquer d'autres unités.

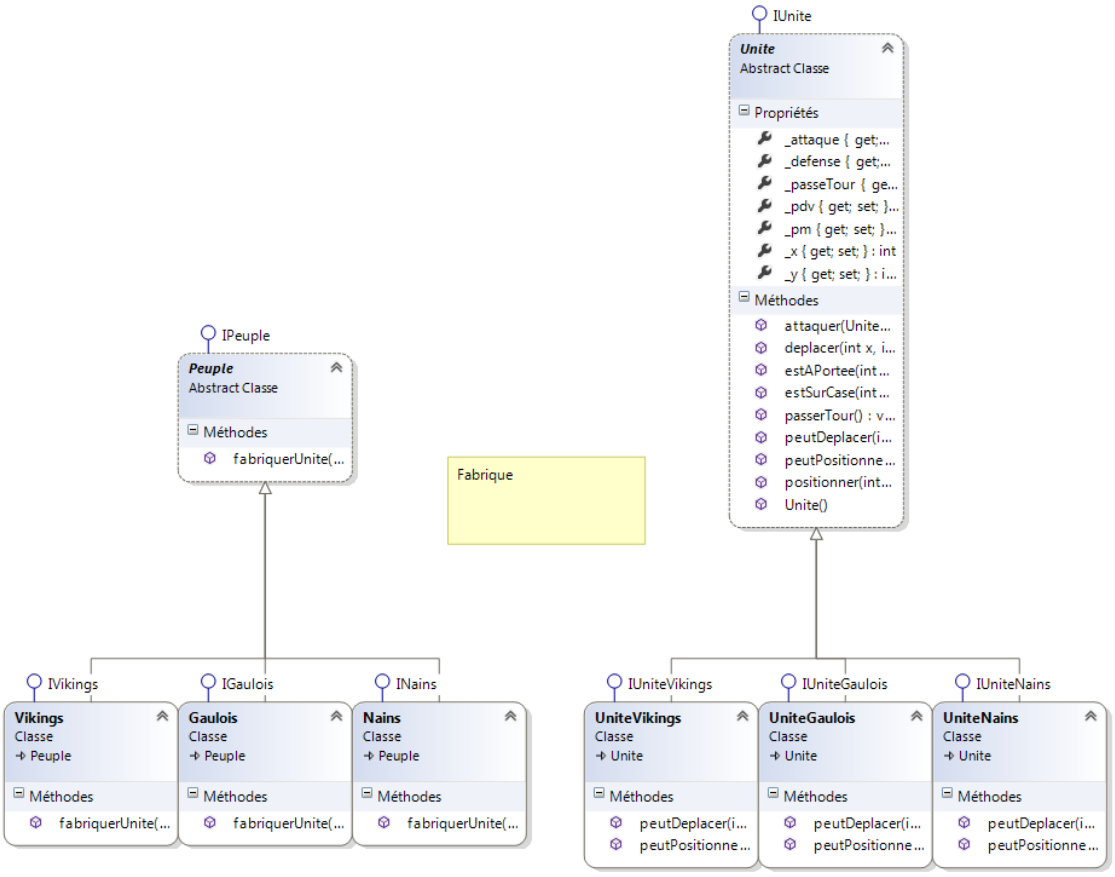


FIGURE 5 – Fabrique

Le deuxième patron utilisé est le monteur. Ce patron permet l'assemblage des différents composants de la partie. La classe CreateurPartie reccueille les informations concernant la création de partie telles que les peuples et la carte choisis puis lance la création de la partie à l'aide du bon monteur. Le monteur assemble les différents composants de la partie (la carte et les joueurs) avant de la renvoyer.

La classe Partie va contenir tout ce qui est utile au déroulement de la partie. Si l'on regarde les attributs, on a les deux joueurs ("_jA" et "_jB"), la carte ("_carte"), le nombre de tour restant ("_toursRestant"), le joueur qui commence ("_premierJoueur"), le joueur qui joue ("_joueur") et l'unité sélectionnée ("_uniteSelectionnee"). Le constructeur de Partie est choisi aléatoirement le joueur qui va débuter la partie. Côté méthodes, on va retrouver les méthodes pour positionner les unités en début de partie("bool peutPositionnerUnites(Joueur j, int x, int y)", "void positionnerUnites(Joueur j, int x, int y)"), une méthode pour passer au joueur suivant ("void joueurSuivant()"), une méthode pour savoir si c'est au tour du joueur A ("bool joueJoueurA()") et les méthodes de selection de cases pour le déplacement et l'attaque ("List<Unite> selectCaseInitiale(int x, int y)", "void selectCaseDestination(int x, int y)").

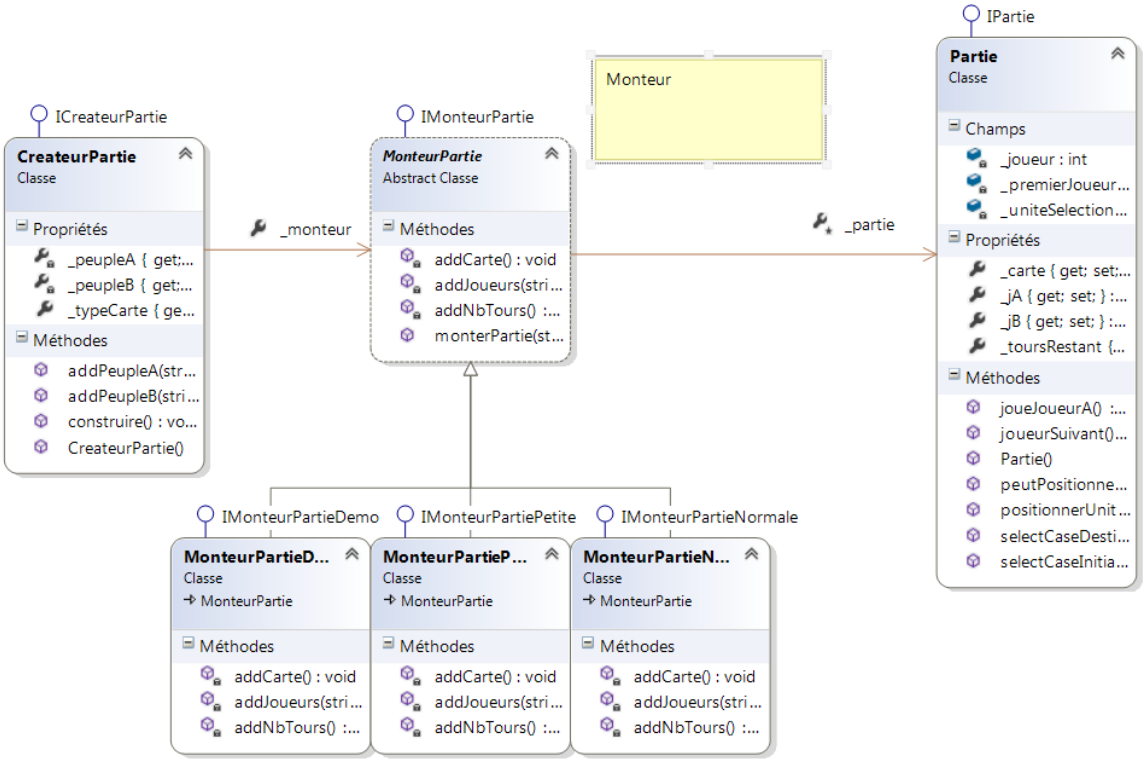


FIGURE 6 – Monteur

Un autre patron utilisé est le poids-mouche. La classe Carte contient un tableau de Cases. Cependant chaque case n'est instanciée qu'une seule fois par la classe FabriqueCase dans l'objectif de ne pas consommer trop de mémoire. L'instance de case souhaitée est donnée par la méthode "Case getCase(string c)".

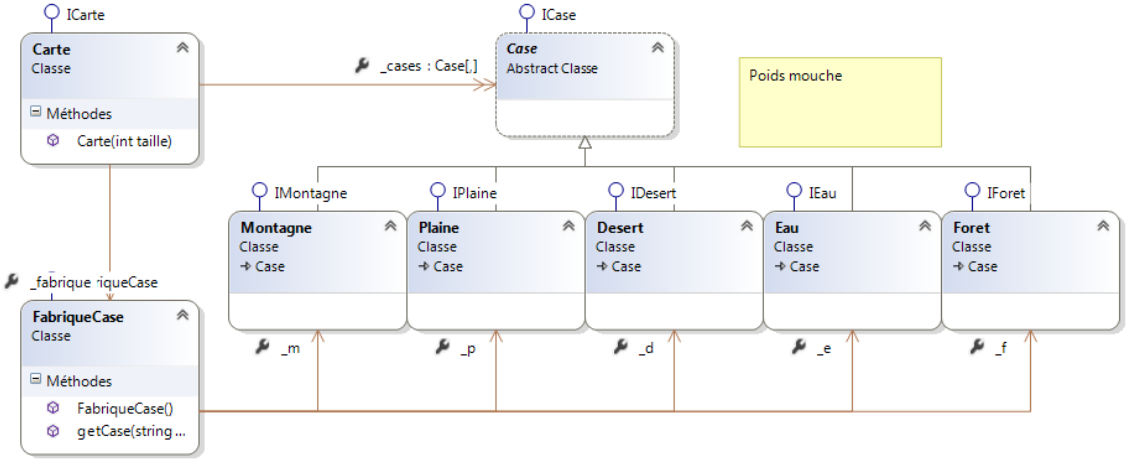


FIGURE 7 – Poids-Mouche

Le dernier patron utilisé est la stratégie. Le créateur carte va faire appel à l'algorythme de création de carte développé en C++ à travaers la méthode "Carte construire()". Mais l'appel a cet algorithme sera différent selon le type de carte à créer. (On rappelle que l'on considère dans notre implémentation trois types de cartes : Démo, Petite et Normale).

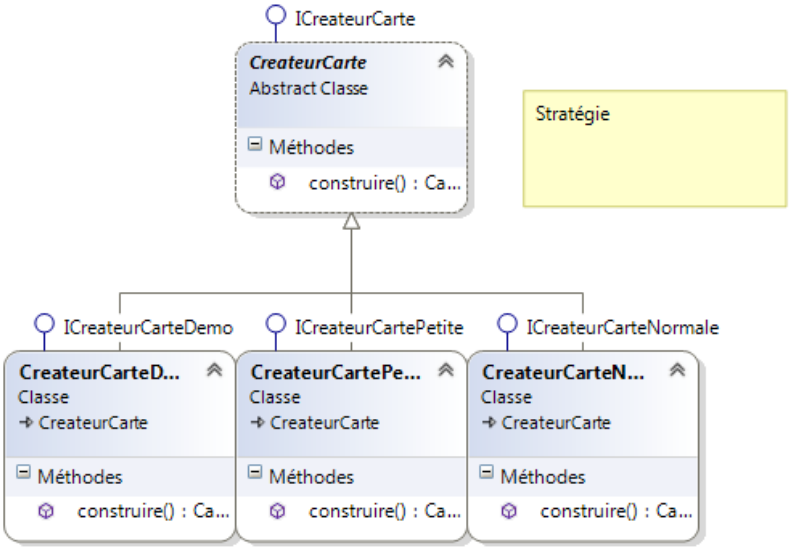


FIGURE 8 – Stratégie

3 Diagramme d'états-transitions

Le diagramme d'états-transitions représenté Figure 9 illustre le cycle de vie d'une unité. Pour commencer, une unité est positionnée sur la carte. Puis durant son tour de jeu, elle pourra soit être sélectionnée puis ensuite être déplacée, attaquer ou passer son tour, soit ne rien faire. Lorsque l'adversaire joue, une unité pourra se faire attaquer. Dans ce cas, elle rentre dans l'état Défense.

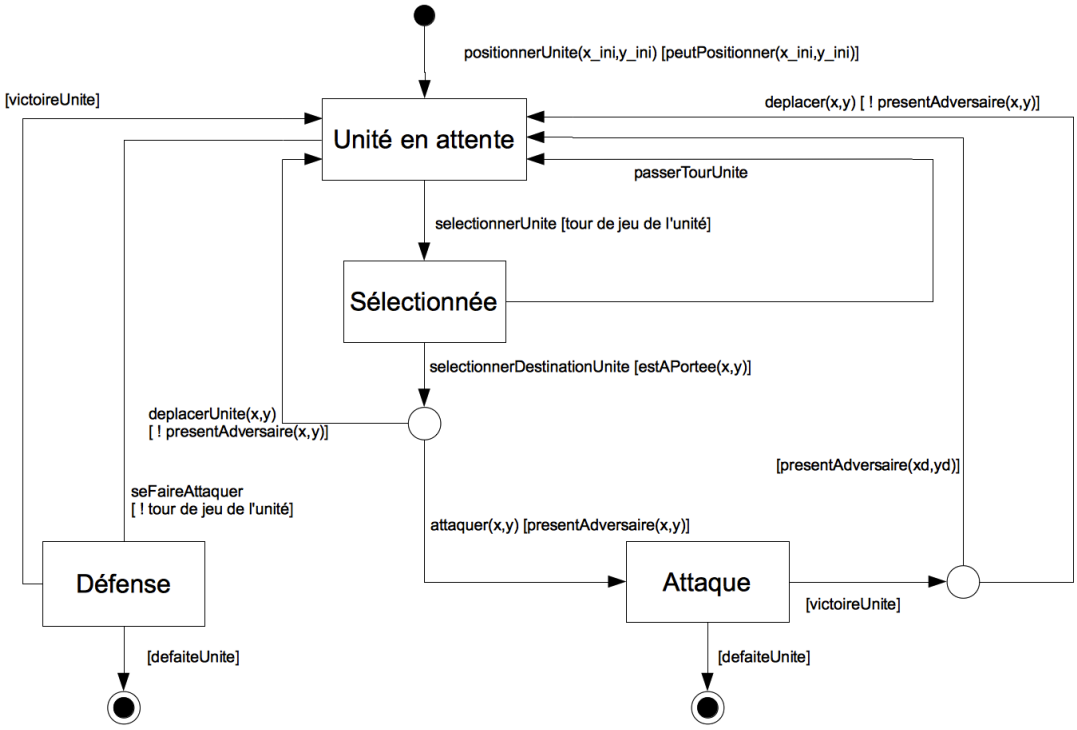


FIGURE 9 – Diagramme d'états-transitions

4 Diagrammes d'intéraction

4.1 Création de partie

En annexe B, on retrouve le diagramme de séquence modélisant la création d'une partie. Pour commencer, le joueur A selectionne la carte et son peuple, puis le joueur B sélectionne son peuple avant que le joueur A lance la création de la partie. Le joueur A ne peut lancer la création de la partie si le joueur B n'a pas choisi un peuple différent du sien.

Puis lors du montage de la partie, trois cas vont se présenter : il faut créer une partie de démonstration, une petite partie ou une partie normale. Pour chaque cas, on va utiliser le monteur adéquat puis lancer le procéssus de création de partie qui consiste à ajouter le nombre de tour, ajouter la carte puis ajouter les joueurs. L'ajout de la carte et l'ajout des joueurs pour la création d'une partie de démonstration sont détaillés dans les Figures 10 et 11

Détailler l'ajout des joueurs ...

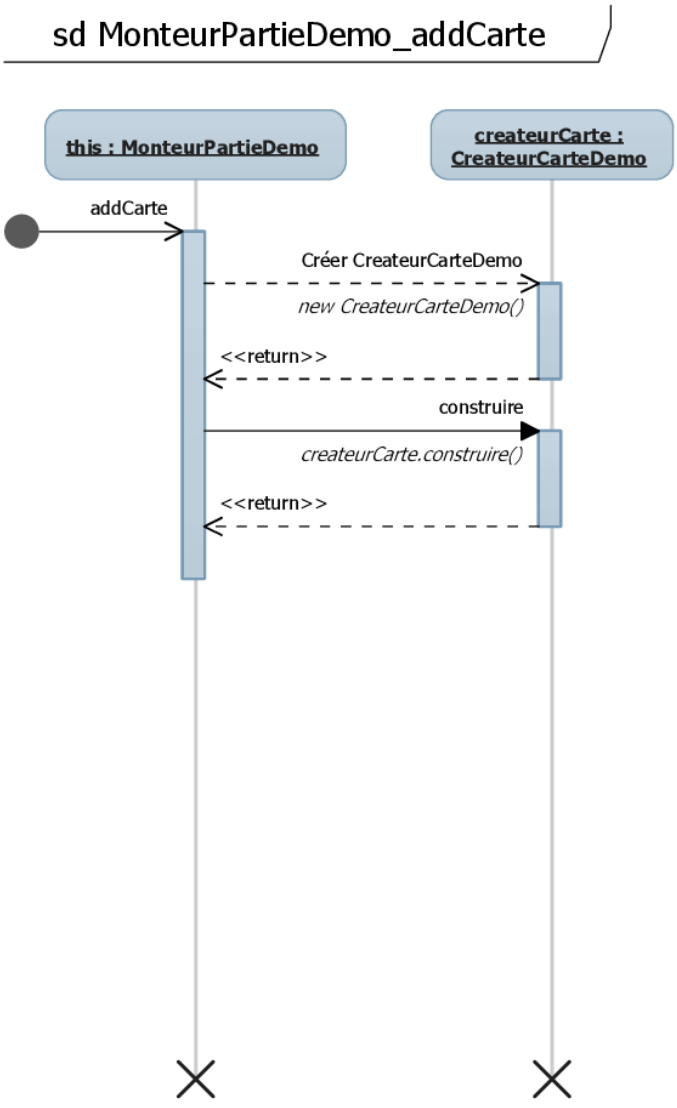


FIGURE 10 – Diagramme de séquence : Ajouter la carte à la partie

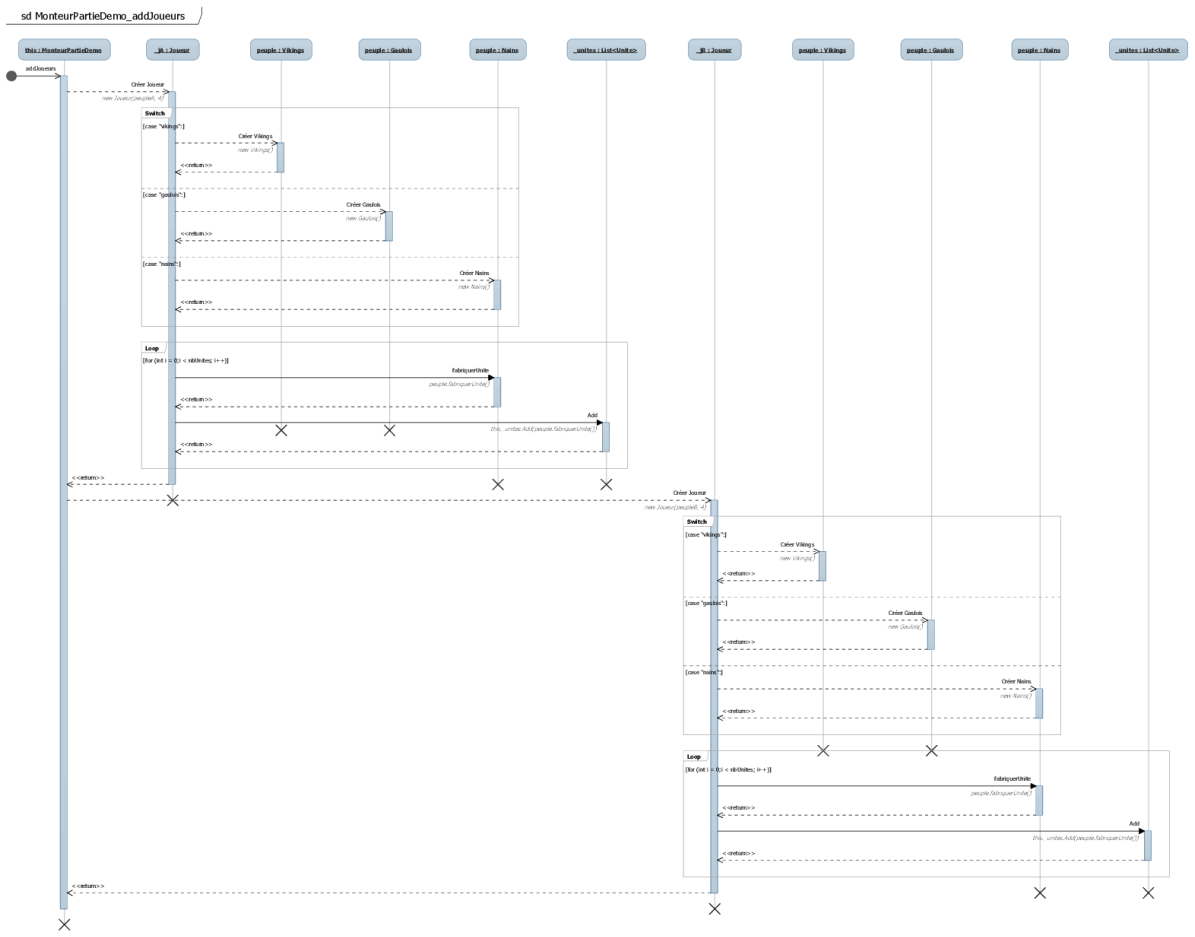


FIGURE 11 – Diagramme de séquence : Ajouter les joueurs à la partie

4.2 Déplacer ou attaquer une unité

En annexe C, on retrouve le diagramme de séquence modélisant le déplacement ou le lancement d'un combat d'une unité. Pour commencer, le joueur selectionne une case. Si ce joueur n'a pas d'unité sur cette case, il ne pourra pas poursuivre cette action.

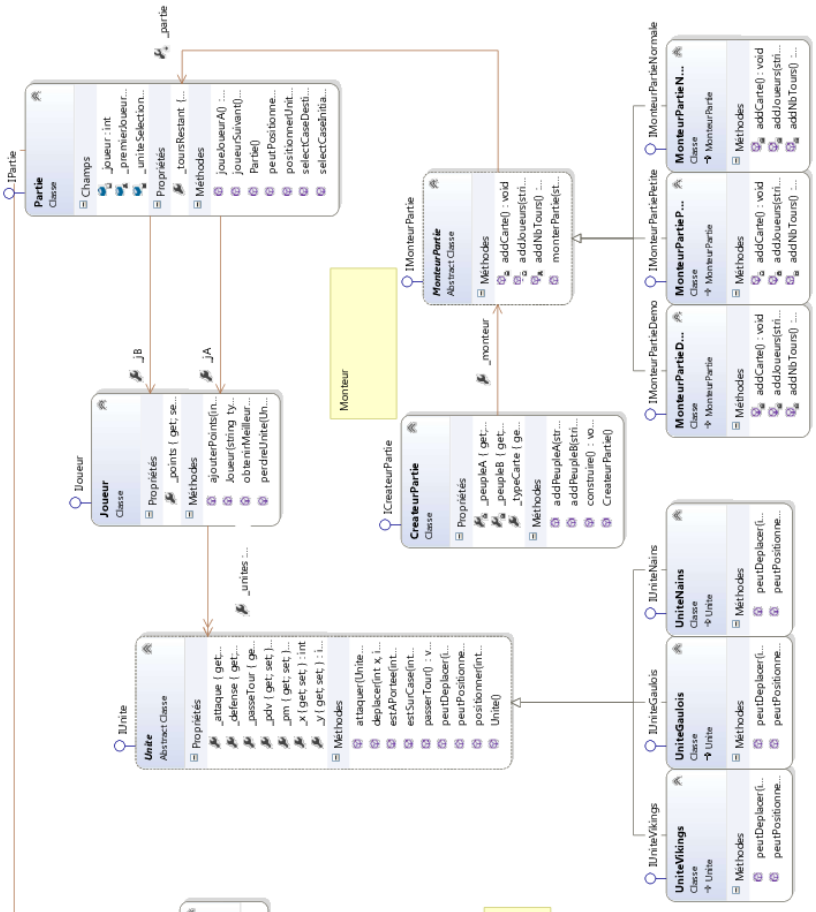
Si, au contraire, le joueur a au moins une unité sur la case sélectionnée, il devra choisir l'unitée à sélectionner et qui sera concervé par l'atribut _unitéSelectionnée de la classe Partie. Ensuite, le joueur devra selectionner une case de destination. On vérifie si cette case est à portée de l'unité sélectionnée, c'est-à-dire si il existe un chemin pour aller jusqu'à cette case sans dépasser le nombre de point de mouvement de l'unité.

Ensuite, soit la case de destination sélectionnée n'a pas d'unités ennemies et l'unité sélectionnée va se déplacer sur cette case, soit il y a au moins une unité ennemie et l'unité sélectionnée attaque la meilleure unitée ennemie présente sur la case.

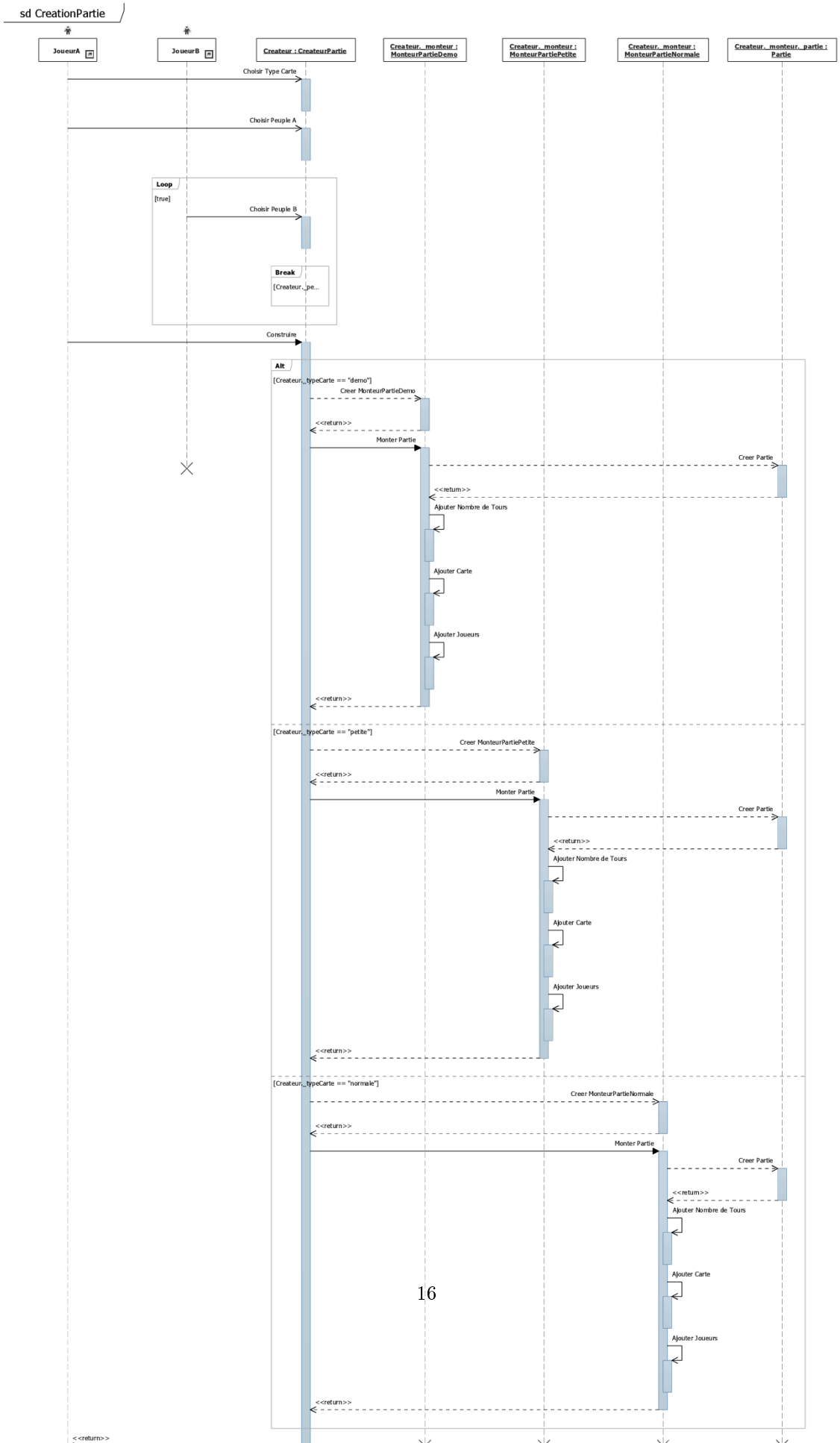
Conclusion

Ce rapport nous donne une idée de la modélisation de notre jeu. Cependant notre modèle n'est pas définitif, il évoluera par la suite avec la phase d'implémentation.

Annexe A Diagramme de classes



Annexe B Diagramme de séquence : Création de partie



Annexe C Diagramme de séquence : Déplacer ou attaquer une unité

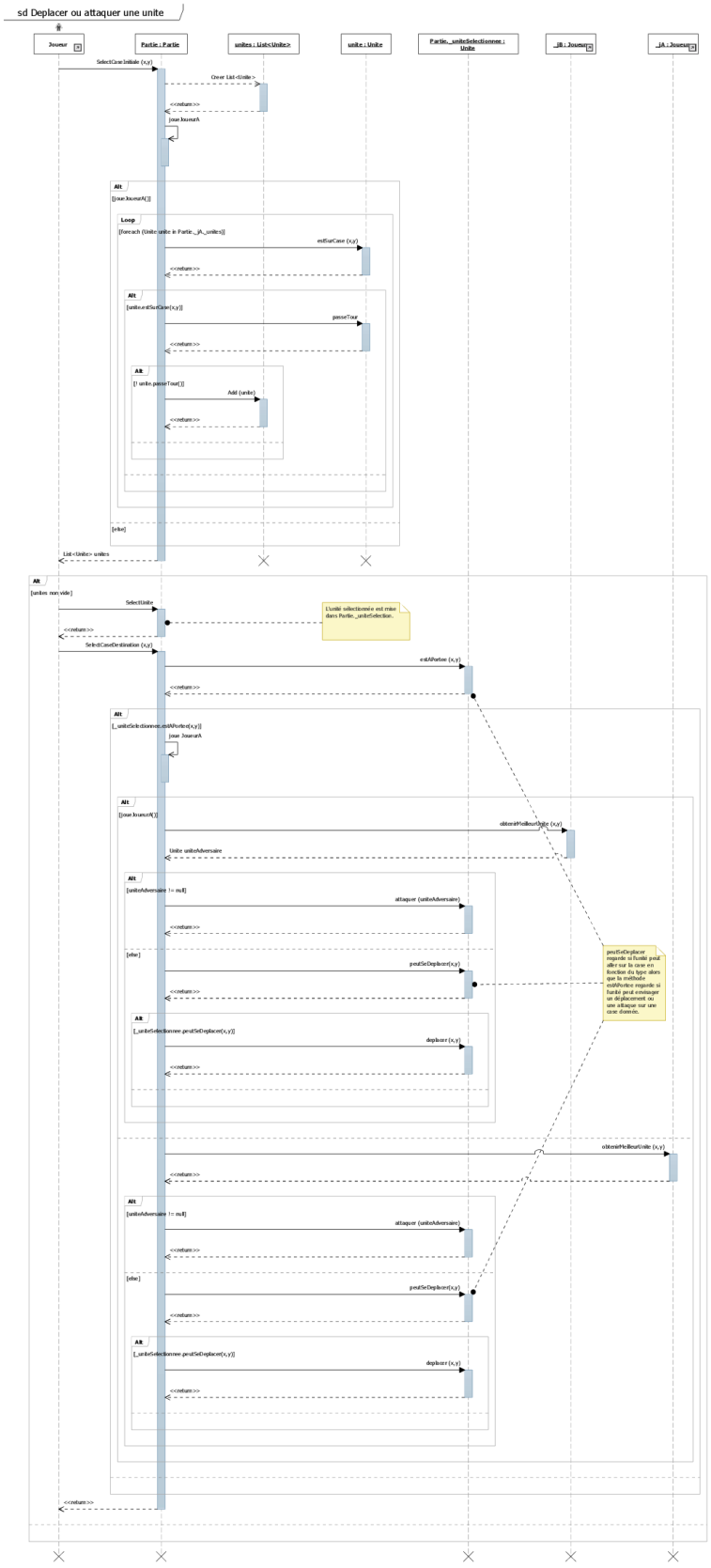


Table des figures

1 Plateau du jeu Small World . . . 3
2 Diagramme de cas d'utilisation global . . . 4
3 Diagramme de cas d'utilisation : Création de la partie . . . 5
4 Diagramme de cas d'utilisation : Tour de Jeu . . . 6
5 Fabrique . . . 7
6 Monteur . . . 8
7 Poids-Mouche . . . 9
8 Stratégie . . . 10
9 Diagramme d'états-transitions . . . 11
10 Diagramme de séquence : Ajouter la carte à la partie . . . 12
11 Diagramme de séquence : Ajouter les joueurs à la partie . . . 13