

Modélisation objet avec UML d'un jeu de billard français

Sommaire

Introduction.....	3
Diagramme des cas d'utilisation	4
Le diagramme de classes.....	5
Cas « lancer une partie ».....	7
Diagramme d'états-transitions.....	7
Diagramme de séquence.....	7
Cas « tirer »	8
Diagramme d'états-transitions.....	8
Diagramme de séquence.....	8
Cas « annuler un coup »	9
Diagramme d'états-transitions.....	9
Diagramme de séquence.....	9
Cas « régler les paramètres d'une partie »	10
Diagramme d'états-transitions.....	10
Diagramme de séquence.....	10
Cas « reprendre une partie »	12
Diagramme d'états-transitions.....	12
Diagramme de séquence.....	12
Cas « sauvegarder une partie »	13
Diagramme d'états-transitions.....	13
Diagramme de séquence.....	13
Conclusion	14

Introduction

Le but de ce projet est de modéliser à l'aide d'UML un jeu de billard français en 3D.

Le billard français est un jeu qui se joue à deux adversaires avec 3 boules en tout (une rouge, une blanche et une pointée). Chaque joueur possède une boule dans laquelle il doit tirer afin de toucher les deux autres boules. S'il arrive à toucher les deux autres boules, il marque un point et rejoue. Sinon, c'est à l'autre de jouer. Le premier arrivé au nombre de points défini en début de partie a gagné.

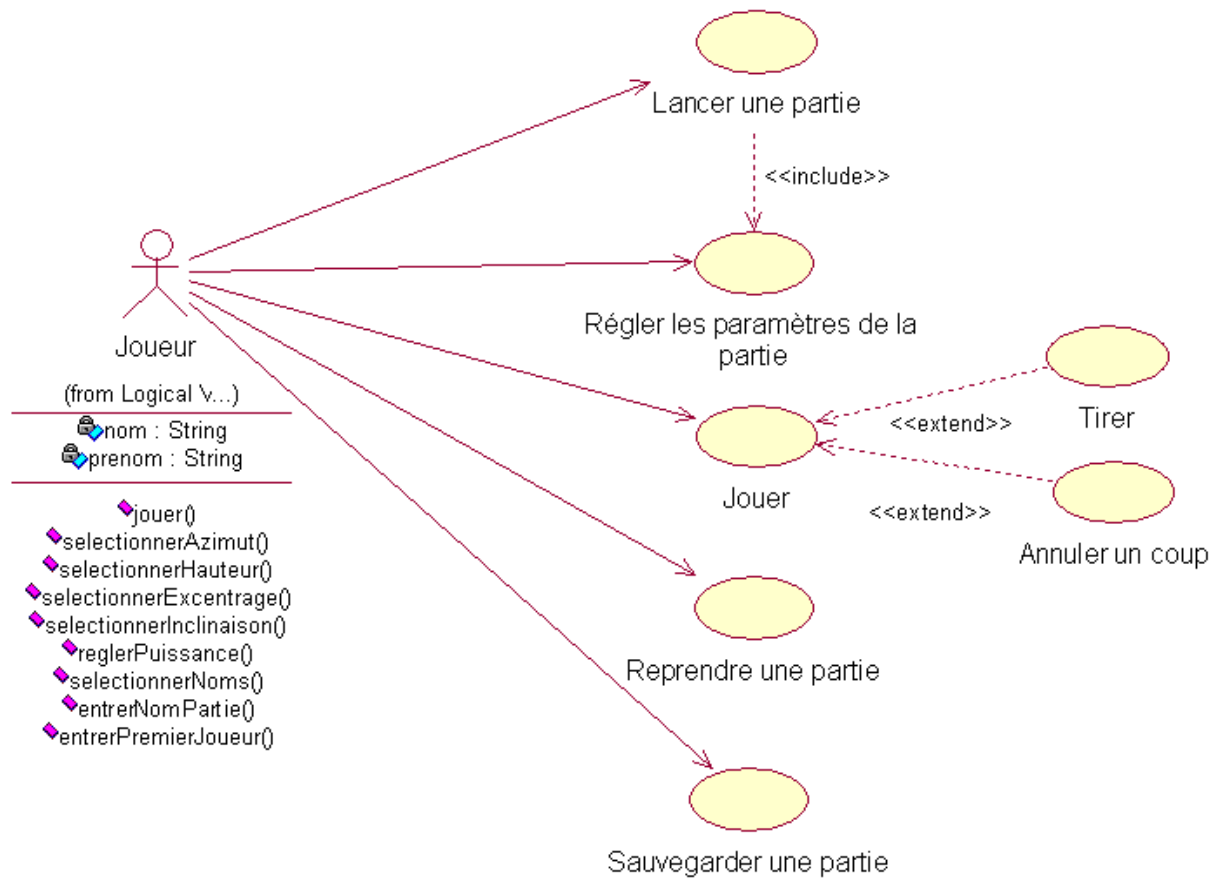
Les paramètres d'un tir sont au nombre de cinq :

- ✓ Azimut : direction du tir (angle entre prolongement de la queue et l'axe x)
- ✓ Angle de hauteur : hauteur de l'impact sur la boule (effets « rétro » ou « coulé »)
- ✓ Angle d'excentrage : position latérale d'impact sur la boule
- ✓ Angle d'inclinaison : inclinaison verticale de la queue lors du tir
- ✓ Puissance du tir (module du vecteur percussion)

La modélisation doit décrire le réglage des paramètres du tir, gérer le tour par tour, compter les points, enregistrer une partie et permettre d'en reprendre une et annuler un coup joué. Sans oublier le réglages des paramètres d'une partie : nombre de rondes, les coefficients de frottement et de rebond, les paramètres de simulation (rapidité de l'animation).

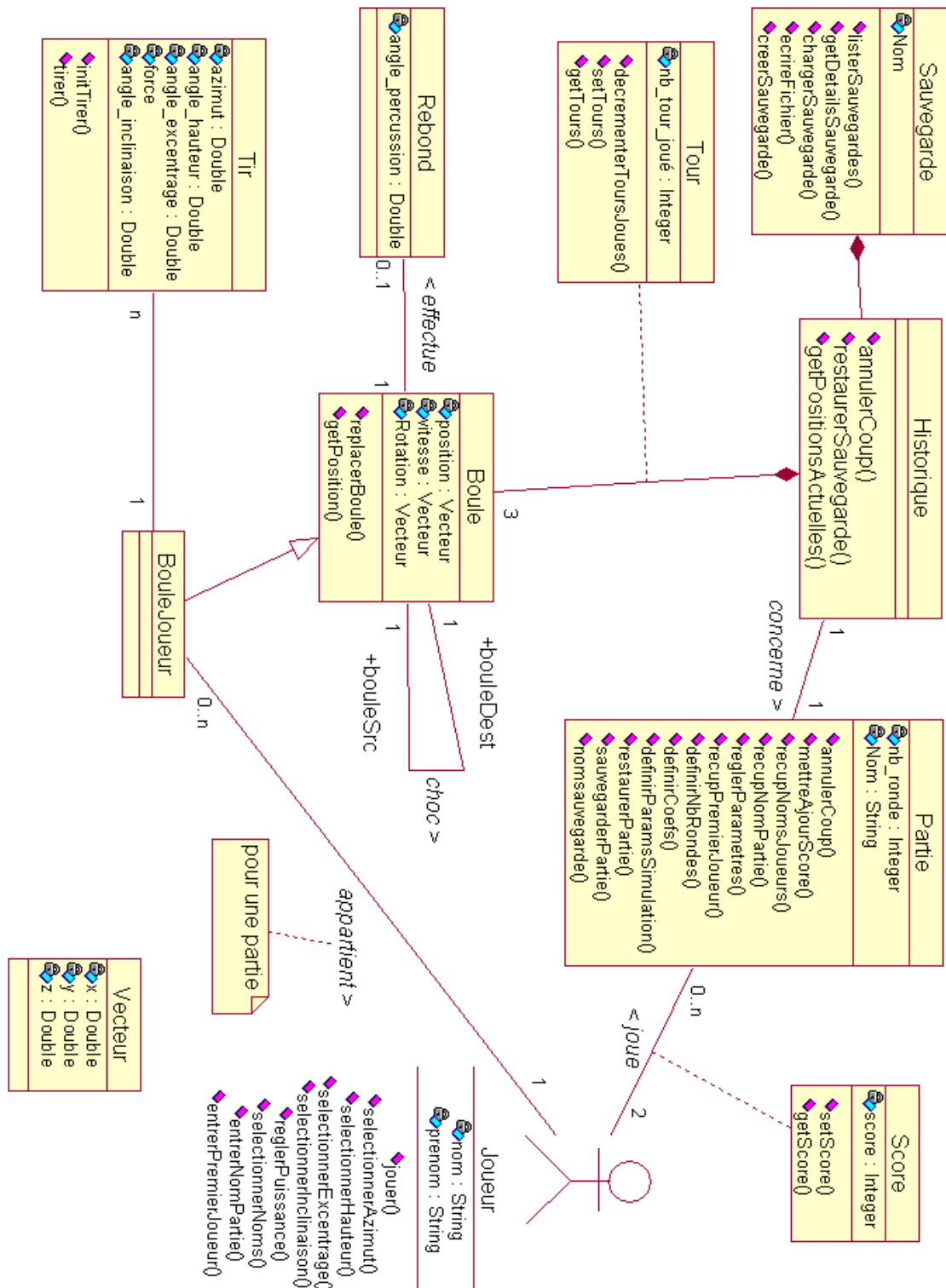
Les modèles UML présentés sont le diagramme des cas d'utilisation, le diagramme de classes, les diagrammes d'états-transitions et diagrammes de séquence pour chacun des cas d'utilisation décrits dans le diagramme des cas d'utilisation. Ces diagrammes ont été réalisées avec l'outil Rational Rose.

Diagramme des cas d'utilisation



Le cas « lancer une partie » appelle obligatoirement le cas « régler les paramètres de la partie » car il est nécessaire de paramétrer une partie avant de jouer. Quand un joueur joue, il peut soit tirer, soit annuler un coup précédent, d'où les cas « tirer » et « annuler un coup » qui étendent le cas « jouer ». Il est également possible de sauvegarder la partie à tout moment, ainsi que de reprendre une partie préalablement sauvegardée.

Le diagramme de classes



La classe *Vecteur* est une classe de confort, pour manipuler directement des vecteurs à 3 dimensions directement, au lieu de modifier les 3 coordonnées des objets à chaque fois (le jeu est en 3D).

Un joueur est représenté par son nom et son prénom. Les opérations décrites lui permettent de jouer un coup et donc également de régler l'azimut de son tir, sa hauteur, son excentrage, son inclinaison et sa puissance avant de tirer. L'opération *selectionnerNoms* permet de renseigner le nom des deux joueurs lors du lancement de la partie. L'opération *entrerNomPartie* permet au joueur de choisir un nom pour la partie et *entrerPremierJoueur* permet de définir quel joueur commencer à jouer.

Il y a 2 joueurs qui s'affrontent dans une partie et chaque joueur possède une boule qui lui appartient pour toute la durée de la partie (*BouleJoueur*).

Une partie est caractérisée par un nom et un nombre de rondes, définis au lancement de la partie. Pour une partie, chaque joueur possède un score (*Score*) qui leur permet de se départager à la fin pour savoir qui a gagné. Les différentes opérations de la classe *Partie* permettent de mettre à jour le score, de régler les paramètres de la partie et de récupérer les informations entrées par l'utilisateur (le joueur) telles que les noms des joueurs, le nom de la partie, le nom de la sauvegarde lors de la sauvegarde d'une partie (objet qui est en relation directe avec le joueur et donc l'entrée de données). Chaque partie dispose d'un historique (*Historique*), qui lui permettent de sauvegarder ou restaurer une partie et d'annuler un coup précédemment joué.

Une sauvegarde (*Sauvegarde*) est une agrégation d'historiques, car elle doit conserver un historique pour chaque coup joué. Elle permet de lister les sauvegardes créées, de récupérer leurs détails, de charger une sauvegarde et de créer une nouvelle sauvegarde pour une partie.

BouleJoueur est une classe qui hérite de *Boule* en ce sens qu'elle ne concerne que les boules appartenant à un des joueurs pour la partie en cours. Une boule est caractérisée par une position, une vitesse et une rotation qui sont toutes des propriétés instances de la classe *Vecteur* (coordonnées 3D). L'opération *replacerBoule* replace la boule dans la position passée en paramètre (instance de *Vecteur* également) et *getPosition* renvoie la position courante de l'instance courante de la classe *Boule*.

Les boules des joueurs (*BouleJoueur*) subissent des tirs, matérialisés par la classe du même nom. Un tir est paramétré par un joueur qui règle son azimut, son angle de hauteur, son angle d'excentrage, son angle d'inclinaison et sa puissance. Ce sont donc naturellement des attributs de la classe *Tir*. L'opération *initTirer* initialise le tir en mettant les valeurs des paramètres du tir par défaut. L'opération *tirer* simule le tir de façon graphique, en fonction des différents réglages du tir.

A partir de ce moment là, chaque boule (*Boule*) peut effectuer un rebond (d'où la nécessité des rôles « source » et « destination » du lien « choc »), caractérisé par son angle de percussion (classe *Rebond*). Le cas où plusieurs rebonds auraient lieu n'est pas représenté, mais on pourrait considérer que l'on simulerait un nouveau tir avec des paramètres de tir différents, calculés à partir du rebond.

L'historique est un agrégat de 3 boules puisqu'il sauvegarde la position de chacune des trois boules du jeu. Le nombre de tours joués (classe *Tour*) est historisé à chaque coup joué, c'est-à-dire sur le lien d'agrégation entre *Historique* et *Boule*.

Cas « lancer une partie »

Diagramme d'états-transitions

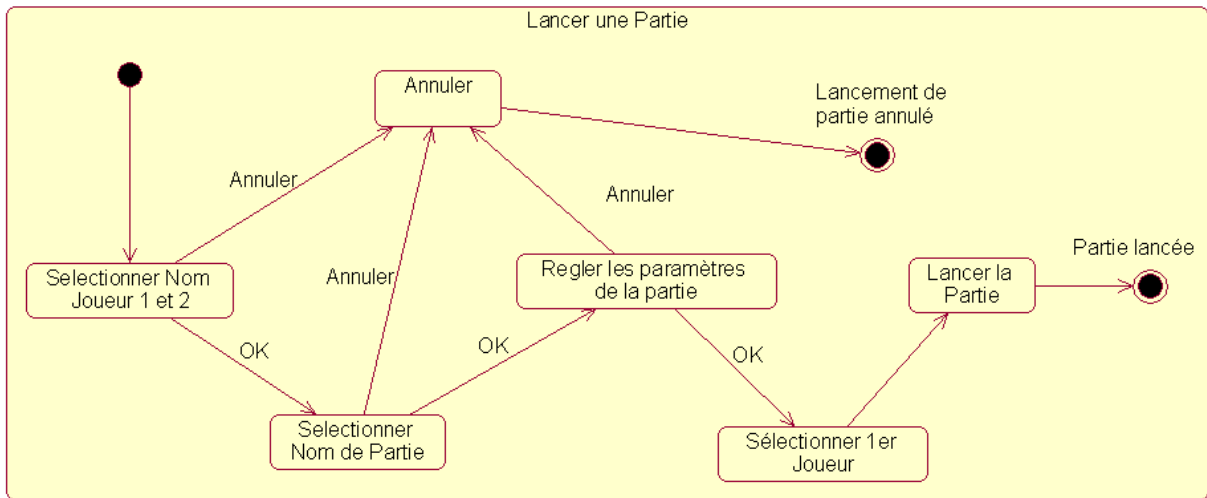
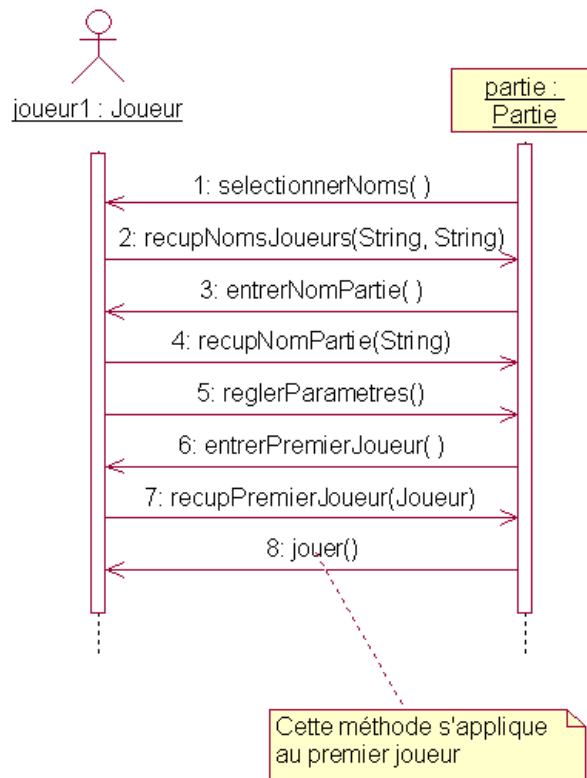


Diagramme de séquence



Cas « tirer »

Diagramme d'états-transitions

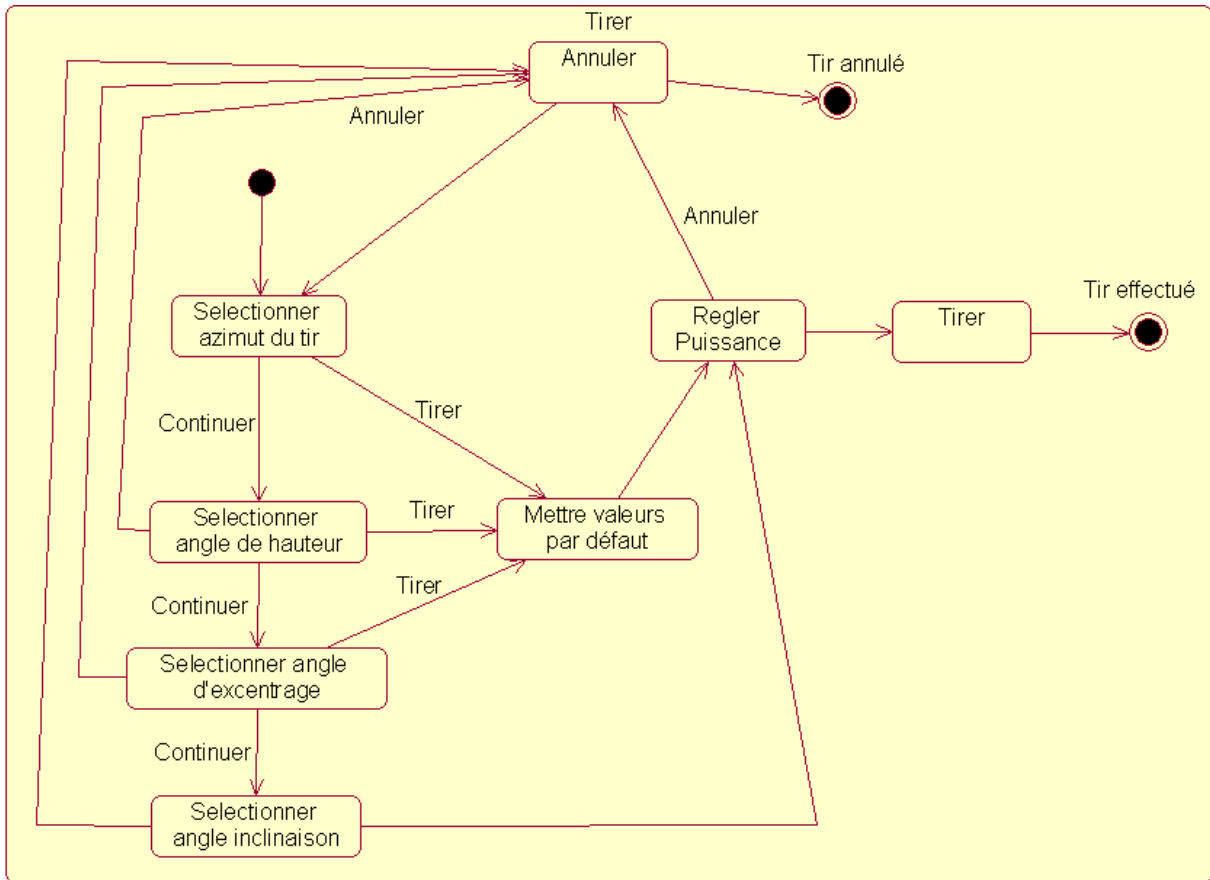
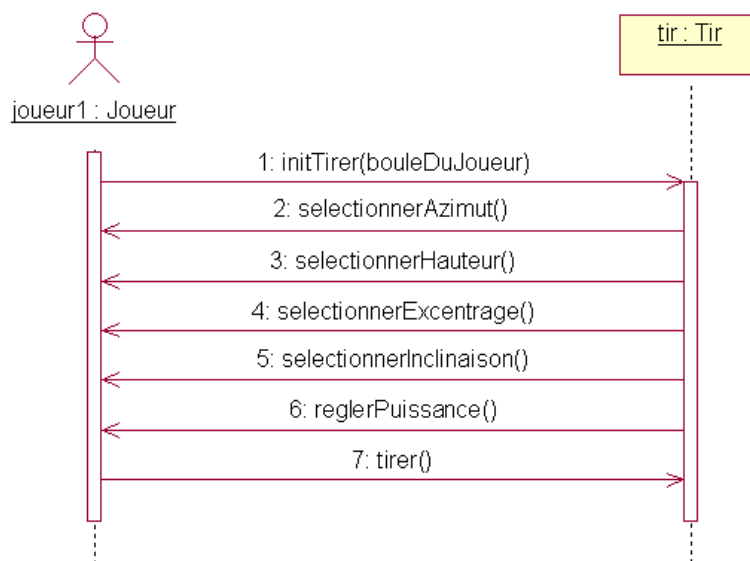


Diagramme de séquence



Cas « annuler un coup »

Diagramme d'états-transitions

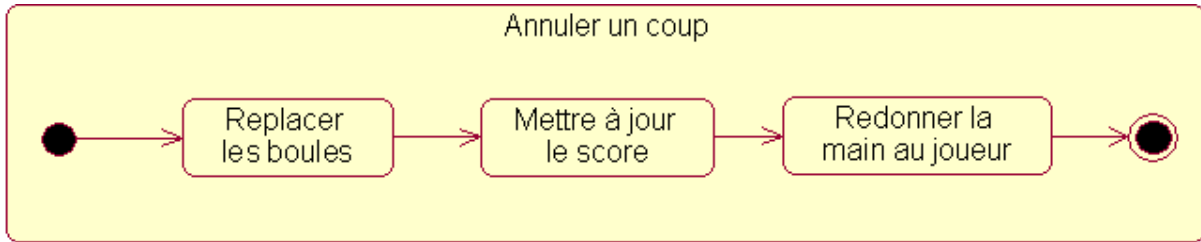
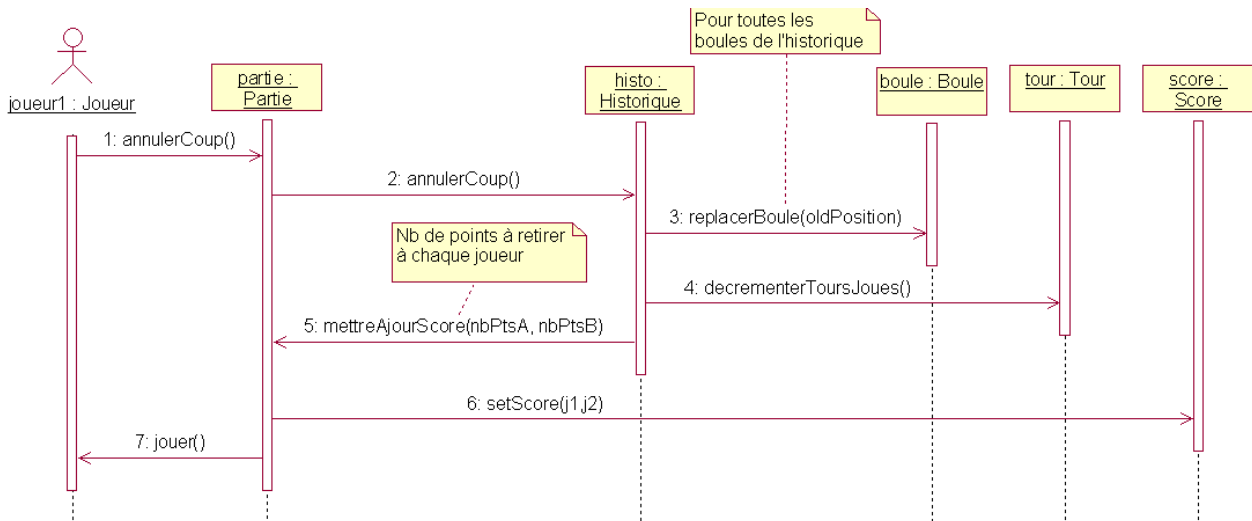


Diagramme de séquence



Cas « régler les paramètres d'une partie »

Diagramme d'états-transitions

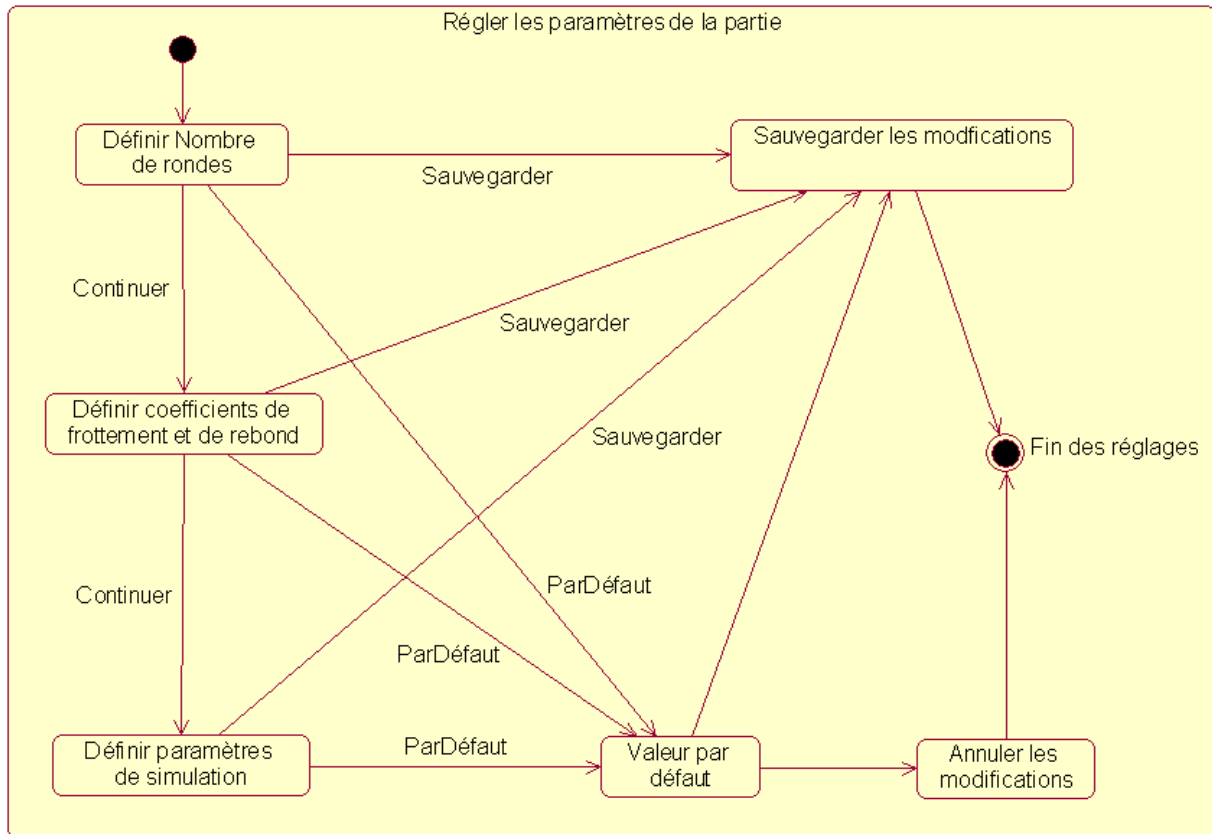
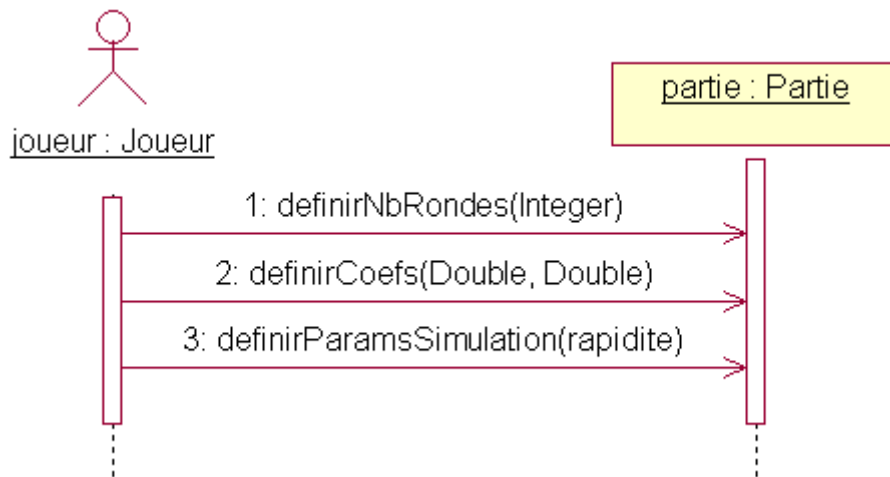


Diagramme de séquence



Cas « reprendre une partie »

Diagramme d'états-transitions

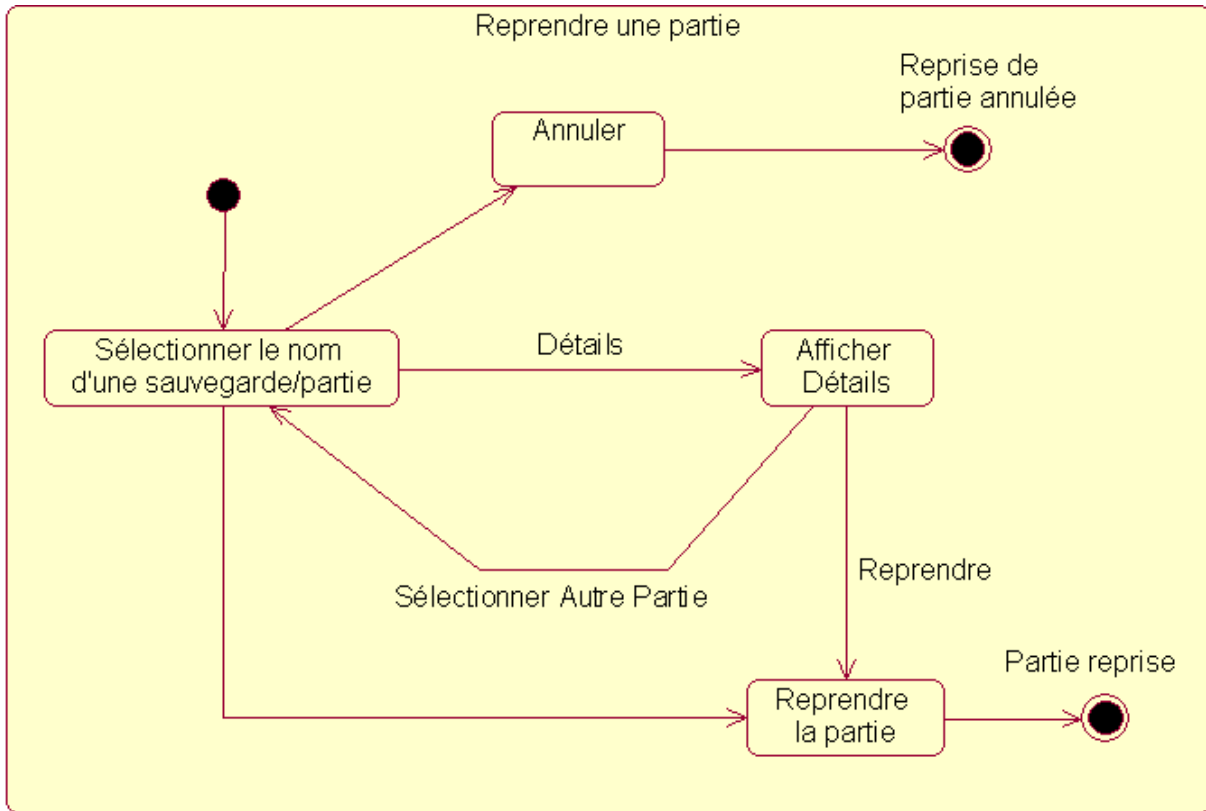
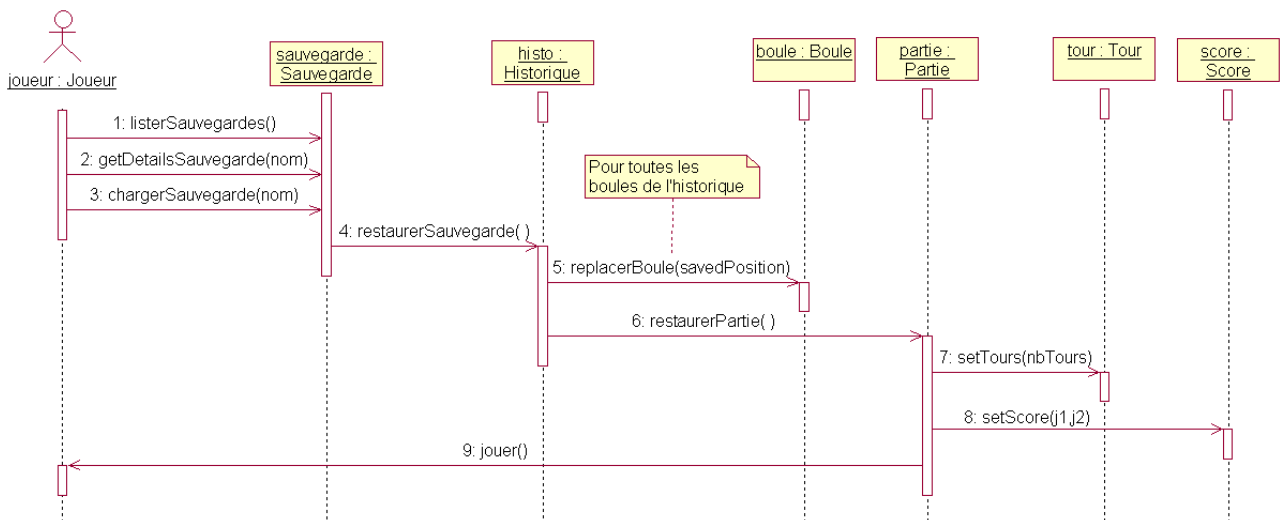


Diagramme de séquence



Cas « sauvegarder une partie »

Diagramme d'états-transitions

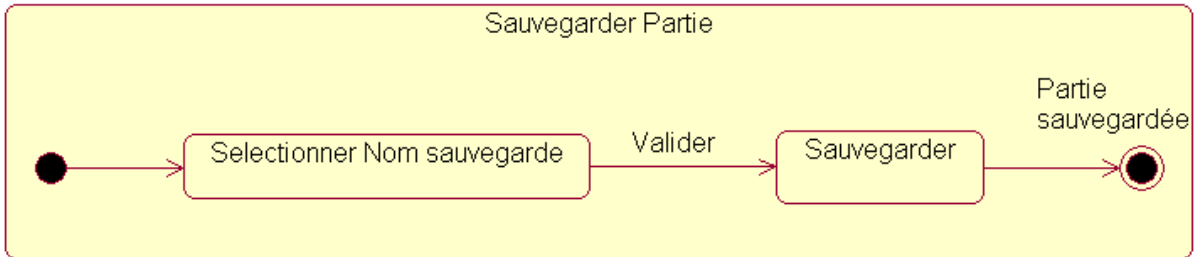
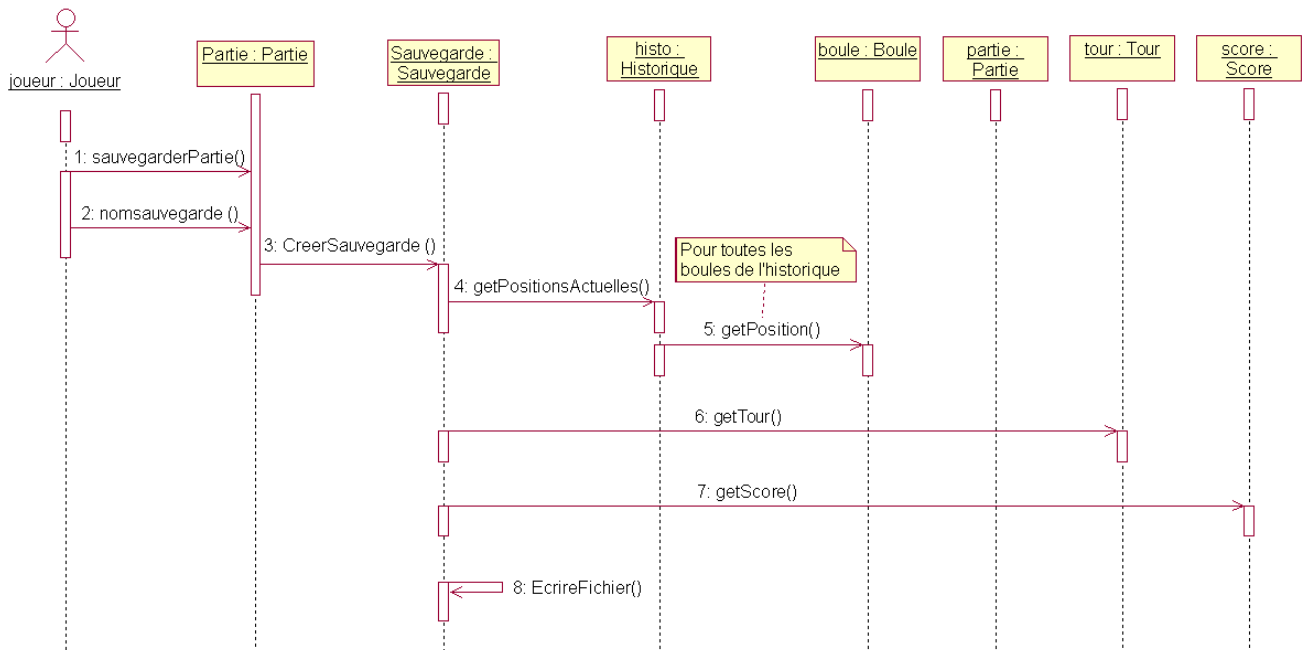


Diagramme de séquence



Conclusion