Université de Technologie de Compiègne

Printemps 2023

LO21 – Programmation et conception orientées objet

Projet SCHOTTEN TOTTEN

Livrable 1 – Prise en main du jeu et analyse des différents concepts

Martin CREUZE, Alexandre LABOURÉ, Adhavane MOUDOUGANNANE,
Jawed GUENCHI, Souhail ELBAAMRANI

Semaine du 27 mars 2023

Ce document synthétise l'ensemble du travail effectué et réflexion menée à la date du 27 mars 2023 sur le Projet SHOTTEN TOTTEN conduit dans le cadre de l'UV LO21 : Programmation et la Conception orientée objet, à l'Université de Technologie de Compiègne durant le semestre de Printemps 2023.

Avant-propos **3**
Introduction **3**
Prise en main du jeu **4**
1.1. Aperçu et but de jeu 4
1.1.1. Schotten Totten 4
1.1.2. Schotten Totten 2 4
1.1.3. Variantes tactiques 5
1.2. Déroulement d'une partie 5
1.2.1. Mise en place 5
1.2.2. Déroulé de la partie 6
1.2.3. Commentaires sur les diagrammes 6
Analyse des différents concepts **7**
2.1. Éléments de jeu 7
2.1.1. Cartes Clan/Siège 7
2.1.2. Tuiles Borne/Muraille 7
2.1.3. Cartes Tactique 7
2.1.4. Chaudron d'huile 7
2.2. Concepts abstraits 7
2.2.1. Joueurs et rôles 7
2.2.2. Mains des joueurs 8
2.2.3. Piles de cartes 8
2.2.4. Partie, tour et manche 8
2.2.5. Combinaisons 8
2.2.6. Édition de jeu 8
2.3. Commentaires 8
Modélisation en classes **9**
3.1. Les classes 9
3.1.1. Gestion des cartes 9
3.1.2. Plateau de jeu 9
3.1.3. Les joueurs 11
3.1.4. La partie 12
3.2. Diagramme de classes 12
Planification des tâches **13**
Bilan **15**

Avant-propos

Dans le cadre de l'UV LO21, nous avons été chargés de concevoir et de programmer une application pour jouer au jeu de société Schotten Totten, conçu par Reiner Knizia. À l'occasion du vingtième anniversaire de ce jeu, Schotten Totten 2 a été créé, ajoutant une dimension asymétrique au jeu original. En outre, il existe plusieurs variantes de jeu (normal, tactique, expert) dans Schotten Totten ou Schotten Totten 2.

L'application doit permettre de jouer à différentes éditions/variantes du jeu sélectionnée par l'utilisateur, incluant à minima la variante de base et la variante tactique de Schotten Totten. Les éléments d'interface graphique utilisateur (GUI) seront développés sous la technologie Qt. L'architecture de l'application doit faciliter l'ajout de nouveaux éléments tels que de nouvelles IA joueurs ou des éléments IHM. Tout cela doit être réalisé sans impacter le code existant, de manière à garantir une évolutivité maximale.

L'objectif du projet est de nous familiariser avec la programmation orientée objet en créant une version numérique du jeu Schotten Totten. Ce faisant, les notions vues en cours de LO21 devront être parfaitement maîtrisées afin de pouvoir être réemployées dans les différentes tâches du projet. Les étudiants doivent, à l'issue de ce projet, prendre conscience de la pertinence de la conception orientée objet pour la programmation de logiciels et de jeux, ainsi que des avantages de l'architecture modulaire pour permettre une évolutivité maximale.

Introduction

Les premières semaines du projet ont permis au groupe de prendre en main les différentes éditions/variantes du jeu Schotten Totten. Nous avons commencé par analyser en détail les règles du jeu et avons joué à plusieurs reprises pour bien comprendre les différentes caractéristiques du jeu.

La cohérence et la robustesse de l'application ne pourront être assurées sans prendre en compte de toutes les spécificités du jeu et la gestion efficace de toutes les interactions entre les entités. Cette phase préliminaire est donc cruciale, car elle permet d'avoir un premier aperçu des entités qui constituent le jeu ainsi que de leur relation. Cela permet ensuite de proposer une modélisation conceptuelle orientée objet du problème, qui servira de base pour l'implémentation du jeu numérique.

Dans ce premier compte-rendu, nous présenterons notre analyse des différents concepts présents dans le jeu et qui devraient apparaître dans l'architecture du projet. Tout d'abord, nous expliquerons brièvement le fonctionnement de Schotten Totten dans chacune de ses éditions/variantes, en détaillant le déroulement d'une partie. Ensuite, nous exposerons les concepts clés qui permettront de définir formellement les classes. Enfin, nous aborderons l'état d'avancement du projet, en décrivant les tâches accomplies et à accomplir, leur planification, leur répartition, ainsi que leur durée estimée.

1
Prise en main du jeu

1.1. Aperçu et but de jeu

Schotten Totten et Schotten Totten 2 sont deux jeux de société qui partagent le même thème et les mêmes mécanismes de base. Cependant, ils diffèrent sur plusieurs aspects du gameplay, tels que les cartes utilisées, le nombre de cartes distribuées, l'utilisation de cartes tactiques ou non, le nombre de bornes/murailles que l'on peut revendiquer à chaque tour, la présence ou non de certains éléments tactiques, etc. Pour mieux comprendre les mécanismes de ces jeux, nous avons examiné les règles officielles de Schotten Totten ainsi que celles de Schotten Totten 2 publiées par le distributeur de jeux de société IELLO.

1.1.1. Schotten Totten

Schotten Totten est un jeu de cartes à deux joueurs qui simule une guerre de territoire entre deux voisins écossais. Le but du jeu est de gagner le contrôle des Bornes en créant des combinaisons de cartes plus fortes que celles de l'adversaire.

Chaque joueur choisit une carte de sa main et la place face visible devant une Borne de son côté de la frontière. Une Borne ne peut contenir plus de trois cartes de chaque côté de la frontière. Une fois qu'une carte est posée devant une Borne, elle ne peut plus être déplacée. Les joueurs piochent ensuite une carte chacun à leur tour, sauf si la pioche est vide, auquel cas ils continuent à jouer sans piocher de cartes.

Après avoir posé une carte sur une Borne et avant de piocher, le joueur peut revendiquer une ou plusieurs Bornes. Pour revendiquer une Borne, il faut obtenir une combinaison de cartes plus forte que celle jouée par l'adversaire. Les combinaisons de cartes sont les mêmes que celles du poker, telles que les suites, les brelans et les couleurs. Une Borne ne peut être revendiquée que lorsque trois cartes sont posées de chaque côté de la frontière.

La partie s'arrête immédiatement si l'un des joueurs contrôle trois Bornes adjacentes ou cinq Bornes dispersées le long de la frontière. Le joueur qui contrôle ces Bornes est déclaré vainqueur.

Schotten Totten est un jeu de stratégie et de chance qui demande de l'observation et de la réflexion pour gagner le contrôle des Bornes.

1.1.2. Schotten Totten 2

Schotten Totten 2 est un jeu de stratégie asymétrique dans lequel deux joueurs s'affrontent : l'Assaillant et le Défenseur. Le but de l'Assaillant est de capturer le château de son éternel rival, tandis que celui du Défenseur est de tenir bon jusqu'à épuisement de son opposant. Pour parvenir à leur objectif, les joueurs doivent élaborer des tactiques efficaces, l'Assaillant devant mener une attaque sur tous les fronts, et le Défenseur devant essayer de résister à toutes les attaques de son adversaire.

L'Assaillant doit endommager quatre murailles ou détruire une muraille en l'endommageant deux fois pour remporter la partie. De son côté, le Défenseur doit tenir bon jusqu'à ce que la pioche soit épuisée, forçant l'adversaire à se retirer.

Pour protéger ses murailles, le Défenseur peut utiliser des chaudrons d'huile bouillante pour neutraliser les attaques ennemies. Les murailles ont une importance majeure dans Schotten Totten 2, car chacune impose des contraintes de pose qui influencent grandement la stratégie de chaque joueur. Le placement des attaques et des défenses doit donc tenir compte de ces contraintes.

Schotten Totten 2 partage de nombreuses mécaniques avec son prédécesseur. Cependant, le jeu comporte plusieurs ajouts tactiques significatifs, ce qui le rend différent et intéressant. Dans ce jeu, les deux adversaires ont des objectifs différents et des armes qui leur sont propres. En choisissant son camp en début de partie, chaque joueur opte pour une stratégie radicalement différente, générant ainsi des sensations de jeu très distinctes. Cette asymétrie est l'une des nouveautés les plus marquantes de Schotten Totten 2, qui se démarque ainsi de la première édition.

1.1.3. Variantes tactiques

Schotten Totten propose des variantes tactiques dans les deux éditions. Ces variantes se jouent avec les mêmes règles de base, mais introduisent des mécaniques supplémentaires apportées par de nouvelles cartes appelées Tactiques.

Pour jouer avec les cartes Tactiques, une nouvelle pioche doit être constituée en mélangeant toutes les cartes Tactiques. Cette pioche est placée à côté de la pioche principale. Il est important de noter que chaque joueur reçoit une carte de Clan/Siège de plus qu'avec les règles de base.

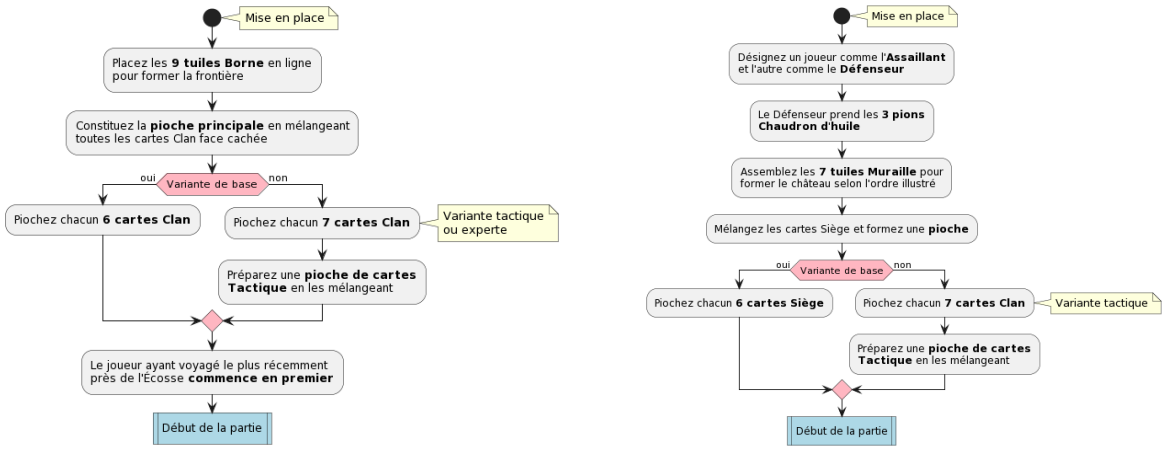
À chaque tour, les joueurs ont la possibilité de jouer soit une carte de Clan, soit une carte Tactique. Les cartes Tactiques ajoutent des capacités spéciales au jeu de base. Chaque carte Tactique possède une capacité différente qui peut influencer le cours de la partie de manière significative. Lorsqu'ils piochent pour compléter leur main à sept cartes, les joueurs peuvent choisir de piocher une carte de Clan ou une carte Tactique.

Schotten Totten 1 propose également une variante experte très similaire à la variante tactique, mais avec une différence importante : les joueurs ne peuvent revendiquer une Borne qu'au début d'un nouveau tour. Cela rend le jeu plus complexe et tactique, car les joueurs doivent planifier leurs mouvements avec encore plus d'attention.

1.2. Déroulement d'une partie

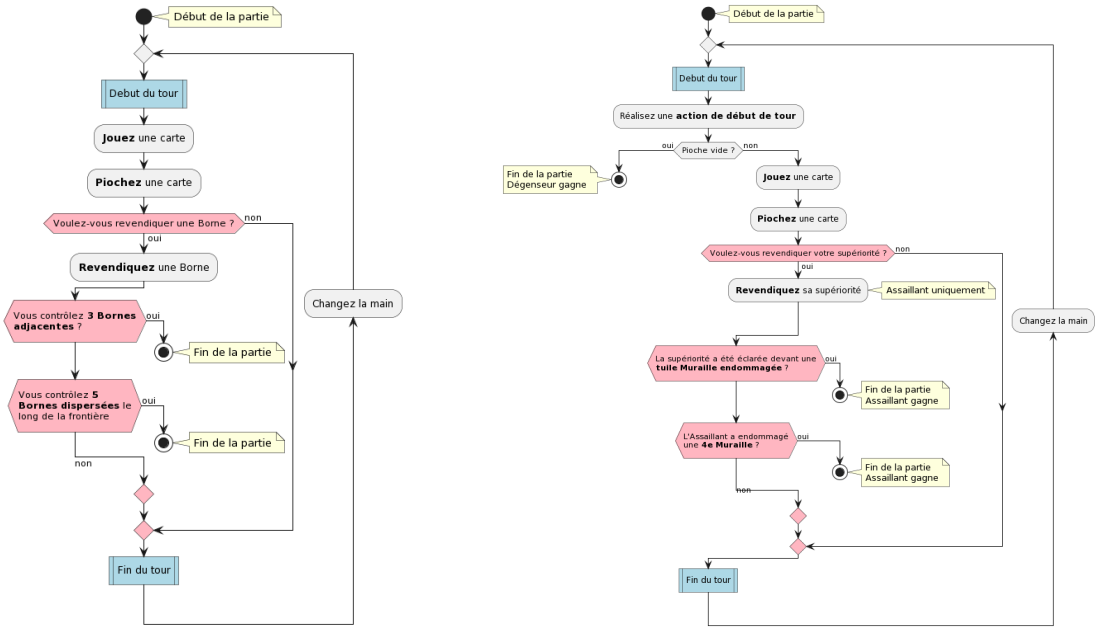
Dans la partie suivante, nous allons présenter le déroulement d'une partie de Schotten Totten et de Schotten Totten 2. Nous ne chercherons pas à paraphraser les règles des deux jeux, qui sont très bien documentées par l'éditeur IELLO. Au lieu de cela, nous nous concentrerons sur les aspects macros les plus importants d'une partie et les fonctionnalités clés à implémenter. Pour faciliter la compréhension du gameplay, nous utiliserons des diagrammes.

1.2.1. Mise en place



Mise en place des parties de Schotten Totten (à gauche) et Schotten Totten 2 (à droite)

1.2.2. Déroulé de la partie



Déroulé des parties de SCHOTTEN TOTTEN (à gauche) et SCHOTTEN TOTTEN 2 (à droite)

1.2.3. Commentaires sur les diagrammes

Les diagrammes présentés ci-dessus fournissent une vue d'ensemble du déroulement d'une partie de SCHOTTEN TOTTEN et SCHOTTEN TOTTEN 2. Ils permettent de comprendre les grandes étapes de la partie sans se concentrer sur les mécaniques de jeu très spécifiques. En effet, il est important de souligner que ces diagrammes ne prennent pas en compte toutes les règles spécifiques qui caractérisent les deux éditions du jeu. D'autres règles très particulières doivent également être prises en compte, mais pour une première approche et afin de décomplexifier les représentations, nous avons choisi de ne pas les aborder ici.

2
Analyse des différents concepts

2.1. Éléments de jeu

Dans la section suivante, nous allons dresser une liste des principaux éléments présents dans les deux éditions de Schotten Totten. Bien que ces jeux diffèrent dans leur *lore* et leur terminologie, ainsi que dans certaines de leurs mécaniques, ils partagent de nombreuses similarités. Parmi ces éléments, on peut citer les cartes Clan/Siège, les Tuiles, les cartes Tactiques, etc. Nous allons ici explorer ces principaux éléments.

2.1.1. Cartes Clan/Siège

Les deux versions de Schotten Totten utilisent des cartes distribuées aux joueurs en début de partie. Dans la version Schotten Totten 1, les joueurs jouent des cartes Clan pour revendiquer une borne du mur. Dans Schotten Totten 2, les cartes Clan sont remplacées par des cartes Siège, mais le principe de jeu reste le même : la combinaison de cartes Clan/Siège jouées détermine si le joueur peut revendiquer une tuile.

2.1.2. Tuiles Borne/Muraille

Les tuiles Borne de Schotten Totten 1 et les tuiles Muraille de Schotten Totten 2 ont toutes les deux un rôle crucial dans leur jeu respectif. Dans Schotten Totten 1, les Tuiles Borne sont plutôt neutres, offrant simplement une position à conquérir pour les joueurs. Dans la seconde édition, les tuiles Muraille ont des propriétés plus fortes qui permettent des mécaniques de jeu différentes. Malgré ces différences, ces tuiles ont en commun le fait qu'elles représentent des positions stratégiques à revendiquer pour gagner la partie. Les deux éléments héritent d'un même concept de tuile, qui est au cœur du gameplay.

2.1.3. Cartes Tactique

Les deux jeux, Schotten Totten 1 et 2, intègrent des cartes tactiques qui portent le même nom et qui ont des fonctions similaires. Ces cartes sont distribuées en début de partie aux joueurs et peuvent être jouées à chaque tour pour influencer les combinaisons de cartes sur les tuiles. Les cartes tactiques sont un élément clé qui instaure une dynamique de jeu similaire dans les deux éditions.

2.1.4. Chaudron d'huile

Schotten Totten 2 intègre un nouvel élément de jeu : le Chaudron d'huile bouillante. Les jetons sont attribués au Défenseur du château, et lui donnent la possibilité de retirer une carte jouée par l'Attaquant devant une des tuiles.

2.2. Concepts abstraits

Un jeu ne se résume pas seulement à des matériaux concrets comme des cartes ou des pièces. En effet, pour jouer, il faut également des joueurs, une organisation des parties, un système de points et de nombreuses autres règles abstraites qui définissent les différentes étapes de la partie. Ces concepts sont souvent difficiles à visualiser et à comprendre, mais ils sont essentiels pour bien comprendre le jeu et y jouer efficacement. Dans cette partie, nous allons examiner de plus près ces concepts et leur intégration dans les règles de Schotten Totten et Schotten Totten 2.

2.2.1. Joueurs et rôles

Le premier concept abstrait à considérer dans un jeu est bien évidemment le joueur. Le joueur est le cœur de toutes les interactions dans le jeu, car sans joueur, il n'y a pas de partie. Dans la première édition, les deux joueurs ont le même rôle et cherchent tous les deux à défendre et à conquérir des sections du mur.

En revanche, dans Schotten Totten 2, les joueurs jouent un rôle différent : l'un est l'Assaillant, et tente de prendre possession du mur, tandis que l'autre est le Défenseur, et essaie de le protéger.

2.2.2. Mains des joueurs

La main d'un joueur est constituée de cartes qu'il peut jouer au cours de la partie. Au début d'une partie, chaque joueur reçoit un nombre de cartes qui varie entre 6 et 7 selon la variante tactique choisie. Au fur et à mesure que la partie progresse, les joueurs piochent de nouvelles cartes pour remplacer celles qu'ils ont jouées. La composition de la main d'un joueur évolue tout au long de la partie en fonction des actions des autres joueurs et des choix qu'il fait lui-même.

2.2.3. Piles de cartes

Le jeu de cartes comprend une pioche principale de cartes Clan/Siège pour revendiquer les neufs cartes Tuile frontalières et pioche de cartes Tactique si la variante jouée le permet. Les joueurs piochent des cartes Clan/Siege pour constituer leur main de départ. En plus des pioches, il y a également une pile de défausse où les cartes sont placées après avoir été jouées ou défaussées. Dans l'ensemble, toutes ces cartes sont empilées sur la table, formant des piles distinctes pour chaque type de carte.

2.2.4. Partie, tour et manche

Une partie est composée de plusieurs tours, chacun constitué d'une séquence d'actions effectuées par les joueurs. Dans Schotten Totten, chaque tour est initié par le joueur qui a la main et se compose de l'action de jouer une carte, suivie d'actions spécifiques liées aux cartes jouées. La partie se termine lorsque l'un des joueurs est désigné comme vainqueur et l'autre comme perdant.

On considère une partie comme un ensemble de manches. À chaque manche, les joueurs accumulent des points en fonction de leur performance. Le vainqueur de la partie est déterminé par le cumul des points à la fin de toutes les manches.

2.2.5. Combinaisons

Dans Schotten Totten, les combinaisons de cartes sont un élément clé du jeu. En effet, les joueurs doivent faire preuve de stratégie et de tactique pour construire les combinaisons les plus efficaces permettant de revendiquer une carte Tuile et ainsi remporter la partie.

2.2.6. Édition de jeu

Une partie utilise une édition spécifique du jeu, comme Schotten Totten 1 ou Schotten Totten 2, ainsi qu'une variante de base, tactique ou experte. Cela définit un concept clé car chaque édition et chaque variante apportent des règles spécifiques qui peuvent considérablement changer la façon dont le jeu est joué. Par conséquent, l'implémentation de ce concept est cruciale car cela permet de proposer des extensions, de nouvelles variantes et de nouvelles tactiques pour le jeu. Cela permet une grande évolutivité et flexibilité du jeu. En outre, cela offre également aux joueurs la possibilité de personnaliser leur expérience de jeu en choisissant l'édition et la variante qui conviennent le mieux à leur style de jeu.

2.3. Commentaires

Au-delà des éléments du jeu, il faut aussi définir les interactions et les relations entre ces différents éléments. Par exemple, les capacités spéciales des cartes tactiques ne sont pas mentionnées dans les concepts évoqués ci-dessus, mais il sera possible d'implémenter ces mécaniques grâce à des méthodes appropriées. Pour l'instant, nous nous concentrons sur la définition des concepts de base de Schotten Totten afin de définir les classes de notre modélisation conceptuelle. Nous aborderons ensuite les relations entre ces classes dans une étape ultérieure.

3
Modélisation en classes

Dans cette troisième partie de notre rapport, nous allons présenter les principales classes qui figureront dans notre modélisation conceptuelle UML pour le jeu Schotten Totten. Ces classes ont été définies à partir de l'analyse approfondie des différents concepts et éléments du jeu que nous avons présentés dans la partie précédente. Cette modélisation permettra d'avoir une représentation visuelle et structurée de l'ensemble des éléments du jeu et de leurs interactions, ce qui facilitera la compréhension et la mise en œuvre de notre solution informatique.

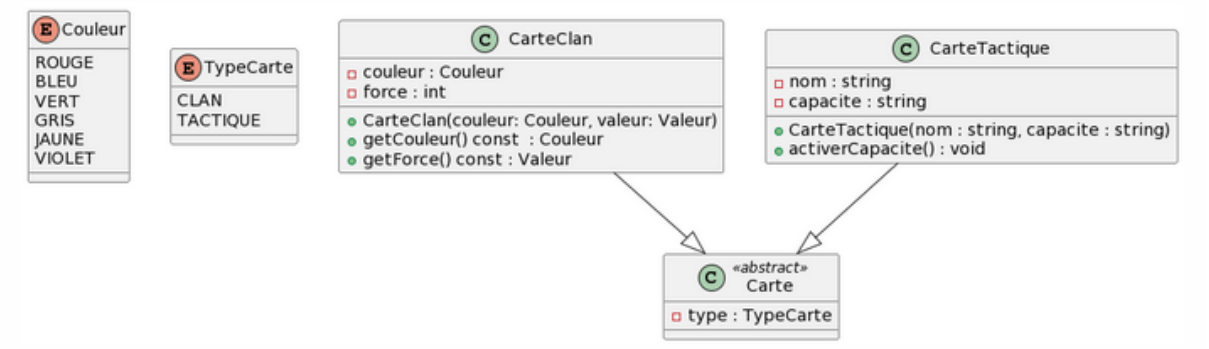
Il est important de garder à l'esprit que la modélisation conceptuelle que nous présentons ici est provisoire et sujette à des modifications et améliorations ultérieures, à mesure que nous continuons à travailler sur notre projet.

La modélisation présentée se concentre exclusivement sur le jeu Schotten Totten et ses trois variantes. Cependant, elle est pensée de manière à faciliter l'intégration d'une éventuelle extension à l'édition Schotten Totten 2.

3.1. Les classes

3.1.1. Gestion des cartes

Les cartes sont de deux types différents : soit des cartes clans soit des cartes tactiques. Les cartes clans sont définis par leur couleur et leur force tandis que les cartes tactiques par leur nom et leur capacité. On obtient alors les classes suivantes.



Chaque carte tactique possède une capacité bien précise. La méthode `activerCapacite()` englobera les actions à réaliser pour chaque type de carte tactique. Par exemple, l'utilisation de la carte Joker ajoutera une carte clan sur une borne choisie par l'utilisateur avec la couleur et la valeur désirée par le joueur, via l'utilisation des méthodes des autres classes.

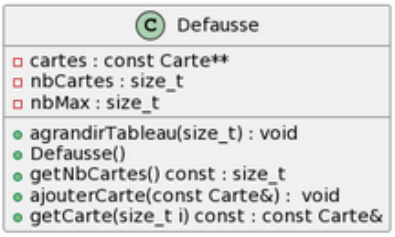
3.1.2. Plateau de jeu

A chaque début de manche, il est demandé de mélanger l'ensemble des cartes et de créer une pile pour constituer la pioche. Il faut également savoir si la pioche est vide, pour arrêter de piocher mais tout de même continuer la partie.

La méthode `piocher()` permet de piocher une carte de la pioche. Elle renvoie la carte piochée et la supprime de la liste de cartes de la pioche.

La méthode `estVide()` renvoie `true` si la pioche est vide, `false` sinon.

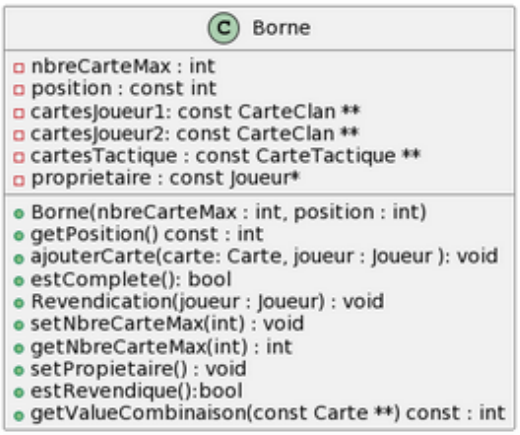
Si l'utilisateur choisit de jouer avec les règles tactiques, nous créerons deux instances de cette classe. Qu'une seule pioche si on joue avec les règles de base.



La défausse est composée d'un tableau de pointeurs vers les cartes. Elle est d'une certaine taille arbitraire (`nbMax`) au départ de 10 cartes par exemple. On augmentera la taille du tableau en fonction du besoin avec la fonction `agrandirTableau()` lors de l'appel de `ajouterCarte()`. L'espace sera alloué dynamiquement.

Pour pouvoir naviguer facilement parmi les cartes de la défausse, qui par ailleurs doivent être accessibles pour les joueurs, nous y ajouterons dans le code un *Iterator*. Nous avons mis en attendant une méthode `getCarte()` pour obtenir la carte de la *i*-ème position.

La classe `Borne` représente les bornes que les joueurs vont tenter de conquérir pendant la partie. On rajoute :

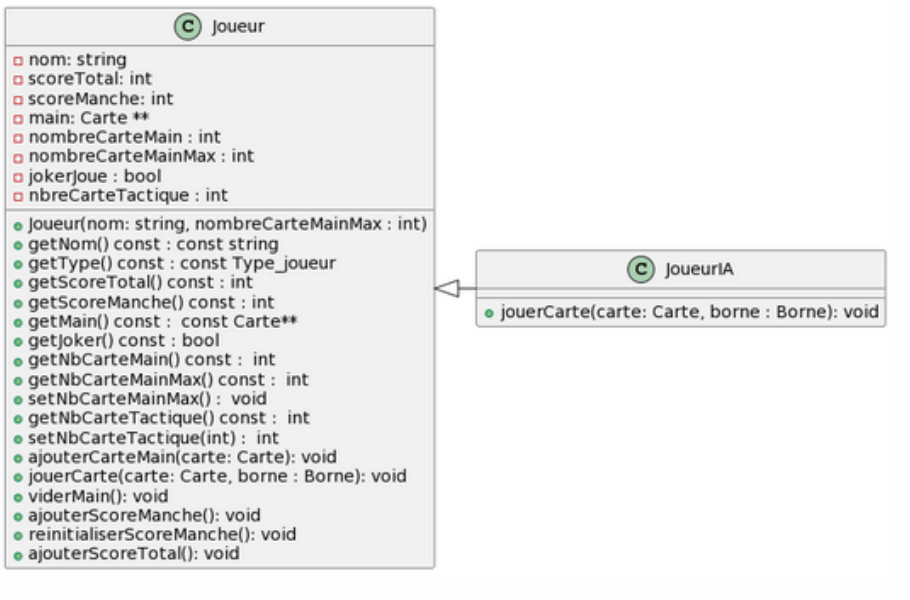


- une méthode permettant de rajouter une carte du côté du joueur qui la pose en vérifiant s'il reste de la place ;
- une méthode permettant de vérifier si la borne est complète, c'est-à-dire si 3 cartes ont été posées des deux côtés (ceci étant la limite). Dans le cas où la carte tactique *X* a été jouée sur la borne, le maximum de cartes pouvant être jouées sur cette borne passe à 4 de chaque côté de la borne ;
- une méthode permettant de trouver si un joueur a réellement la possibilité de revendiquer la borne lorsqu'il fait la demande : il faut vérifier que la méthode `estComplete()` retourne vrai ou qu'il n'existe aucune option de combinaison plus forte pour l'adversaire, puis vérifier si sa combinaison est en effet plus grande que celle de l'adversaire. Si c'est le cas, on fait ensuite appel à `setPropiétaire()` ;
- `estRevendique()` permet de savoir si la borne est revendiquée ou non. Si elle l'est, il est plus possible de poser de cartes dessus ;
- `getValueCombinaison()` renvoie la valeur de la combinaison des cartes du côté de la Borne souhaitée (1 : somme, 2 : suite, 3 : couleur, 4 : brelan et 5 : suite couleur)

Il existe une `piocheTactique` et une `defausse` seulement dans la variante.

3.1.3. Les joueurs

Nous avons pour objectif de créer une IA capable de jouer au jeu. Il faut alors distinguer le statut du joueur entre un humain et une IA.



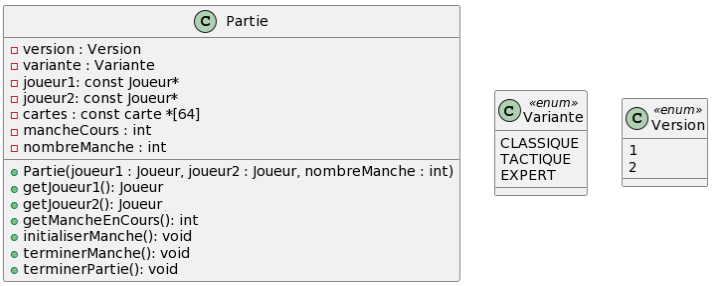
La classe `Joueur` représente un joueur de la partie. Il possède un nom/pseudo, un type (humain ou IA), un score pour la manche en cours, un score dans la partie totale et une main. La main est un ensemble de cartes. On ajoute le constructeur (qui initie le nom, le type, les différents scores à 0, et crée un tableau avec des pointeurs qui pointent sur les pointeurs correspondants aux cartes), le destructeur et les accesseurs. On ajoute les méthodes suivantes :

- dès qu'il pioche, il faut lui rajouter une carte dans sa main ;
- dès qu'il joue/pose une carte, il faut lui supprimer la carte de la main et la dépose sur une borne ;
- une méthode qui vide la main du joueur (pour chaque nouvelle manche) ;
- une méthode qui ajoute un point au score de la manche après chaque revendication ;
- une méthode qui réinitialise le score de la manche dès qu'une nouvelle manche commence ;
- une méthode qui ajoute en fin de manche le score de celle-ci au score total.

À noter également que dans la variante Tactique, il faut connaître si le joueur a déjà joué son joker ou non, et combien de carte tactique il a joué pour ne pas dépasser le nombre plus un de carte tactique joué par son adversaire. Le nombre de cartes maximum dans la main est également différent selon la variante choisie.

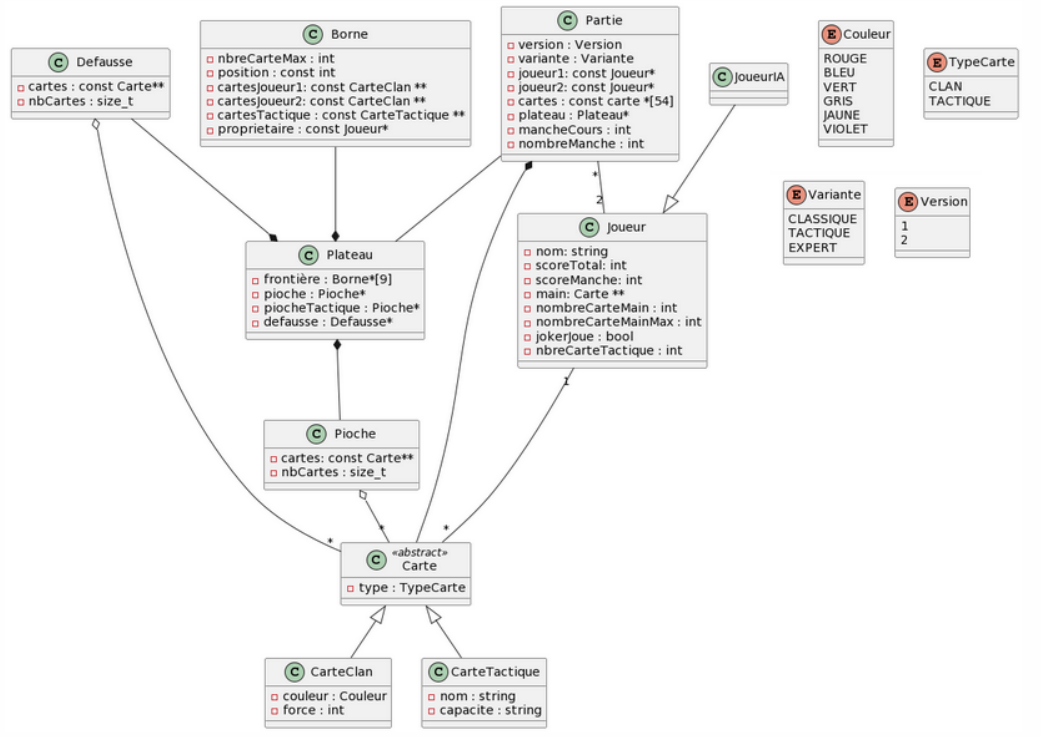
Nous avons créé une classe `JoueurIA` comme une classe héritant de Joueur pour pouvoir créer une méthode `jouerCarte()` qui est propre à l'IA.

3.1.4. La partie



- `initialiserManche()` fait appel à `viderMain()`, à `reinitialiserScoreManche()` de `Joueur`, crée une nouvelle pioche, supprime les propriétaires des bornes ;
- `terminerManche()` permet de connaître quand la manche est finie : c'est-à-dire si un joueur a revendiqué 5 bornes ou 3 bornes adjacentes. Si c'est le cas, il appelle `initialiserManche()` ;
- `terminerPartie()` permet de clôturer la partie.

3.2. Diagramme de classes



UML représentatif des relations entre nos classes (sans les méthodes)

Explication Une partie est composée par un ensemble de cartes. Elle confronte deux joueurs (humain ou IA) qui vont alors combattre sur un plateau de jeu : composé d'une défausse, de pioches et d'une frontière (ensemble de bornes). Selon l'édition et la variante choisie, il existe différents types de carte : les cartes Clans et les cartes Tactiques pour la première édition. Au cours du jeu, celles-ci se retrouvent soit dans les mains des joueurs, soit dans la/les pioches ou soit dans la défausse.

4
Planification des tâches

Afin de mener à bien ce projet, il est important de savoir ce qu'on doit effectuer, et comment l'effectuer. Le projet d'implémentation du jeu Schotten-Totten s'articule en une partie orienté objet effectuée en C++ et en une partie de mise en place graphique avec le logiciel Qt. Les tâches à effectuer sont les suivantes :

- Créer les différentes classes principales et les différentes méthodes associées en C++ :
    - Cartes : Clan, Tactique
    - Borne, Pioche, Défausse
    - Plateau
    - Joueur, JoueurIA (de bas niveau)
    - Partie
- Etudier les différents design patterns et les appliquer sur nos classes si nécessaire
- Effectuer des tests afin de déterminer le bon fonctionnement de ces classes en jeu
    - Créer un programme de test de jeu afin de jouer en mode console
    - Débugger les éventuelles erreurs dans les classes/méthodes
- Retravailler les différentes classes en vue des tests précédents
    - (Facultatif) Implémenter une IA "intelligente" faisant office de second joueur
- Se familiariser avec l'interface graphique Qt
    - Commencer avec une prise en main du logiciel et de ses fonctionnalités
    - Effectuer plusieurs micros-projets en vue de l'implémentation du jeu
- Implémenter Schotten-Totten dans le logiciel Qt
    - Créer une fenêtre graphique (Plateau, boutons, icônes)
    - Implémenter les concepts clés (Borne, Carte, Pioche, etc...)
    - Relier l'interface graphique avec le programme effectué précédemment
    - Implémenter toutes les fonctionnalités restantes
- Tester le jeu

D'autres étapes vont venir s'ajouter au fur et à mesure de la création du jeu, cette liste est non-exhaustive. Tableau de la planification des tâches sur la page ci-après.

Nr	Tâche	Sous-tâches	Priorité	Complexité	Durée totale	Membre (durée)	Date fin	Dépendance	Avancement
1	Livrable 1	Prise en main du jeu	indispensable	facile	5h	Adhavane (5h)	01/04	Indépendante	Effectué
2	Livrable 1	Analyse des différents concepts	indispensable	facile	5h	Adhavane (5h)	01/04	Indépendante	Effectué
3	Livrable 1	Première modélisation en classes	indispensable	difficile	30h	Martin (15h) Alexandre (10h) Adhavane (5h)	01/04	Dépendante des concepts clés	Effectué
4	Livrable 1	Planification du projet	important	facile	5h	Jawed (3h) Alexandre (2h)	01/04	Indépendante	Effectué
5	Livrable 2	Deuxième modélisation en classes	indispensable	difficile	20h	Tout le groupe	20/05	Dépendante des retours	En attente des retours
6	Livrable 2	Création en C++ des classes Cartes	indispensable	moyen	1h	Jawed, Souhail	20/05	Dépendante de la modélisation	Non effectué
7	Livrable 2	Création en C++ des classes Borne, Pioche, Défausse	indispensable	moyen	4h	Jawed, Martin	20/05	Dépendantes de la classe Carte	Non effectué
8	Livrable 2	Création en C++ de Joueur,JoueurIA	indispensable	moyen	3h	Adhavane,Souhail	20/05	Dépendantes de la classe Carte	Non effectué
9	Livrable 2	Création en C++ de la classe Plateau	indispensable	moyen	1h	Adhavane	20/05	Dépendante des classes Carte,Joueur	Non effectué
10	Livrable 2	Test de l'implémentation	indispensable	important	1h	Jawed, Souhail	20/05	Dépendante des classes	Non effectué
11	Livrable 2	Retravail de l'implémentation	indispensable	important	15h	Alexandre, Martin	20/05	Dépend des tests	Non effectué
14	Livrable 3	Prise en main et Implémentation basique dans l'Interface Qt	indispensable	important	15h	Tout le groupe	10/06	Indépendante	Non effectué
15	Livrable final	Implémentation avancée Qt	bonus	important	30h	Tout le groupe	18/06	Dépendante de l'implémentation basique	Non effectué
16	Livrable final	Développer IA "intelligente"	bonus	important	20h	A déterminer	18/06	Dépendante des stratégies	Non effectué
17	Livrable final	Test du jeu	indispensable	important	2h	Tout le groupe	18/06	Dépendante du jeu	Non effectué
18	Livrable final	Retravail du jeu et finalisation	indispensable	important	20h	Tout le groupe	18/06	Dépendante du jeu	Non effectué
19	Livrable final	Vidéo de présentation et commentaires audio	indispensable	important	5h	Souhail	18/06	Dépendante du jeu finalisé	Non effectué
20	Livrable final	Rapport	indispensable	important	20h	Tout le groupe	18/06	Indépendante	Non effectué

Bilan

La cohésion de groupe est un élément crucial pour le succès d'un travail en groupe. Il a été difficile de démarrer ensemble sachant que chacun devait comprendre le jeu de son côté et se faire une idée de comment l'implémenter. Il est possible que nous ayons rencontré quelques difficultés au début du projet en partie dues à un manque de communication. Cependant, nous avons finalement réussi à mettre toutes nos idées en commun et à les expliquer dans ce rapport, ce qui témoigne d'une capacité à communiquer efficacement et à travailler ensemble pour résoudre les problèmes.

Nous allons nous répartir les tâches pour la suite du projet afin que chacun puisse travailler de son côté sur la partie du projet qu'il aura choisie. Nous mettrons ensuite notre travail en commun et chacun expliquera aux autres comment il a réalisé son travail. Ceci permet d'avoir le point de vue des autres personnes du groupe et d'apporter des améliorations au projet.

Nous sommes donc confiant quant à la suite de ce projet et avons hâte de pouvoir tester le jeu final.