

# Jeu d'échec sur Unity

**Dossier de projet - TPI**

Écrit par : **Arthur Bottemanne**

---

Chef de projet : **Loïc Viret**

Premier expert : **Roberto Ferrari**

Deuxième expert : **Claude-Albert Muller Theurillat**

## Glossaire :

**Pièce clouée** : Au échecs, une pièce clouée est définie comme une pièce menacée ne pouvant pas se déplacer sans exposer une autre pièce de plus grande valeur à une capture.

**Moteur de jeu** : Un moteur de jeu est un logiciel qui facilite le développement de jeux vidéo en mettant à disposition des outils tels que la simulation de physique, le rendu graphique, la gestion des entrées utilisateur, et autres fonctionnalités.

**Unity** : Unity est un moteur de jeu multiplateforme en deux dimensions et en trois dimensions.

Table des matières

Glossaire : .....	2
1 Analyse préliminaire.....	4
1.1 Introduction.....	4
1.2 Objectifs.....	4
1.3 Planification initiale.....	6
2 Analyse / Conception .....	7
2.1 Concept.....	7
2.1.1 Diagrammes de flux.....	7
2.1.2 Maquettes.....	12
2.2 Stratégie de test.....	14
2.3 Risques techniques .....	14
2.4 Planification détaillée.....	15
2.5 Dossier de conception.....	20
2.5.1 Diagramme de classes .....	20
2.5.2 Diagrammes de séquences .....	21
3 Réalisation .....	23
3.1 Dossier de réalisation.....	23
3.2 Description des tests effectués.....	23
3.3 Erreurs restantes .....	24
3.4 Liste des documents fournis.....	24
4 Conclusions .....	24
5 Annexes .....	25
5.1 Résumé du rapport du TPI / version succincte de la documentation.....	25
5.2 Sources – Bibliographie.....	25
5.3 Journal de travail .....	25
5.4 Manuel d'Installation.....	25
5.5 Manuel d'Utilisation .....	25
5.6 Archives du projet .....	25

# 1 Analyse préliminaire

## 1.1 Introduction

---

Ceci est le dossier de projet pour mon TPI qui est réalisé dans le cadre du CPNV. Pour ce projet, 90 heures sont mises à disposition pour effectuer ce projet.

L'objectif est de créer un jeu d'échecs représenté en deux dimensions en utilisant Unity comme moteur de jeu. Les règles officielles devront être implémenter et fonctionnel, pour but que deux joueurs s'affrontent dans une partie.

Les parties seront chronométrer ou le temps dépendra du choix des joueurs, et les parties seront enregistrer dans un fichier en notation algébrique

## 1.2 Objectifs

---

L'objectif de ce projet est de développer le jeu des échecs en deux dimensions à l'aide du moteur de jeu Unity.

Le jeu permettra de jouer aux échecs à deux sur le même ordinateur, incluant les coups spéciaux, il permettra aussi de fournir le chronomètre pour chaque joueur afin de pouvoir faire des parties « officielles » ainsi que des parties rapides.

Le jeu doit permettre à deux joueurs de s'affronter aux échecs avec les règles officielles incluant :

1. Ce sont toujours les blancs qui commencent. Les joueurs choisiront eux même qui commence entre les deux.
2. Chaque joueur doit être capable de ne bouger que ses propres pièces
3. Il doit être possible d'activer/désactiver la prévision d'où pourront se déplacer les pièces en fonction de leur type.
4. Les coups spéciaux doivent être implémentés (Roque, prise en passant et promotion)
5. Un joueur ne doit pas pouvoir déplacer une pièce clouée
6. Les conditions de victoire doivent être implémentées (échec, échec et mat, pat, 50 coups, triple répétition, accord et temps)
7. Un fichier de récapitulation de la partie en notation algébrique résumera la partie.

Pour les règles du jeu d'échecs, le candidat peut se baser sur cette page :  
[https://fr.wikipedia.org/wiki/R%C3%A8gles\\_du\\_jeu\\_d%27%C3%A9checs](https://fr.wikipedia.org/wiki/R%C3%A8gles_du_jeu_d%27%C3%A9checs)

Pour la notation algébrique, le candidat peut se baser sur cette page :  
[https://fr.wikipedia.org/wiki/Notation\\_alg%C3%A9brique](https://fr.wikipedia.org/wiki/Notation_alg%C3%A9brique)

Pour les objectifs, il existe aussi 7 points techniques qui seront évalués pour ce projet.

1. Déplacement des pièces

- Respect des différents déplacements des pièces
- Prévision des positions possibles lors d'un déplacement
- Ergonomie du déplacement

2. Gestion du tour des joueurs

- Identification du joueur à qui est le tour
- Possibilité de ne déplacer que ses pièces
- Pièce clouée

3. Coups spéciaux

- Le petit roque
- Le grand roque
- La prise en passant

4. Promotion

- Le déclenchement de la promotion
- Les possibilités de promotion
- L'ergonomie de la fonctionnalité

5. Conditions de victoire

- La gestion de la mise en échec
- La validation de l'échec et mat
- Le pat

## 6. Le fichier récapitulatif

- Les code de pièces
- Les cases concernées
- Les prises

## 7. Setup fonctionnel avec son protocole d'installation

- Sans erreur sur une machine Windows 10 64bits
- Pertinence du nom de l'exécutable et de son emplacement
- Protocole d'installation clair

## 1.3 Planification initiale

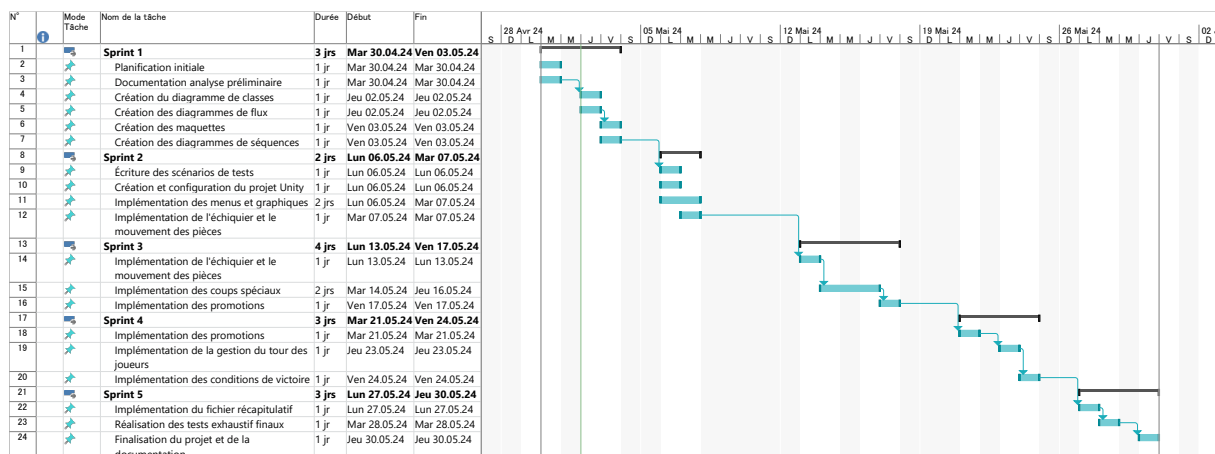


Figure 1: Planification initiale

La planification initiale a été réalisé en planifiant autours du nombre de jours qu'une tâche prendrait à finir pour avoir un aperçu plus général. Cela veut dire qu'une tâche pourrait être indiqué qu'elle prend une journée à compléter, mais soit en réalité plus rapide à faire.

Les tâches de tests et de documentation sont aussi prises en compte lors des tâches d'implémentation, car j'ai jugé que durant l'implémentation, les documentations et les tests nécessaire seront fait simultanément.

Les tâches de tests et de documentation seront planifiées durant la révision de la planification initiale.

## 2 Analyse / Conception

### 2.1 Concept

---

#### 2.1.1 Diagrammes de flux

---

Pour cette conception, cinq diagrammes de flux ont été réalisés pour démontrer la logique des aspects importants du programme.

Les diagrammes de flux représentent respectivement :

- La gestion du mouvement des pièces
- La gestion des mouvements spéciaux (roque, en passant)
- La gestion du tour des joueurs
- Les conditions de fin de partie
- La gestion du fichier récapitulatif

Avec ces diagrammes de flux, on peut avoir un aperçu entier sur la logique du programme.

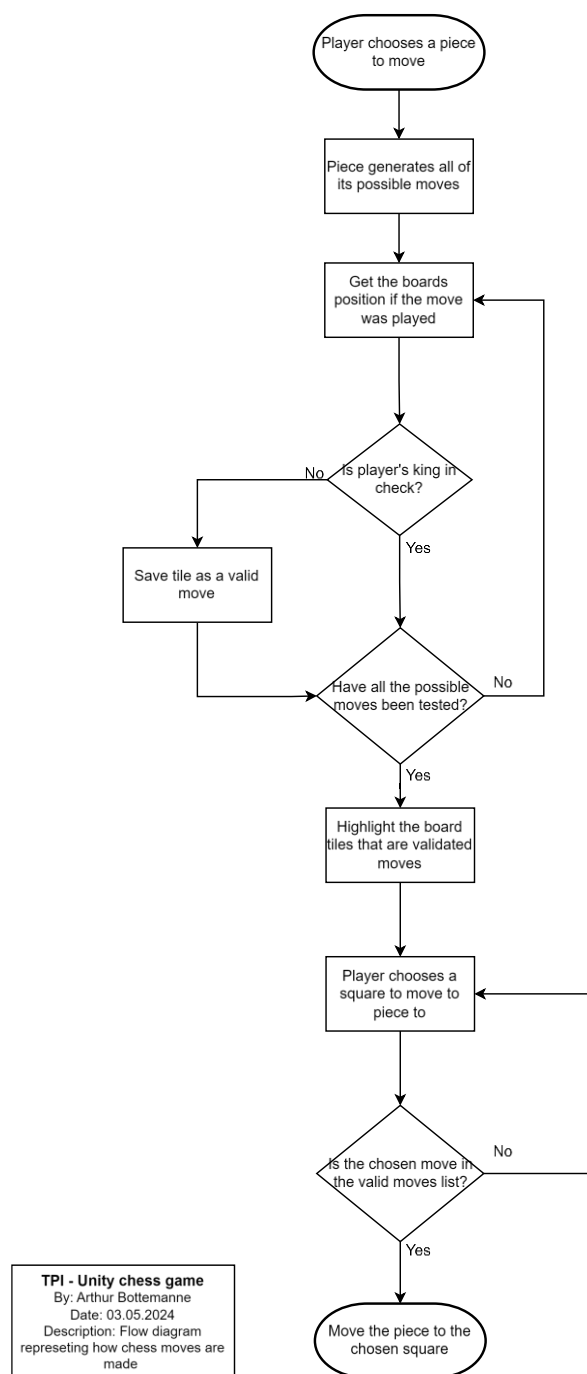


Figure 2: Diagramme de flux de la gestion du mouvement des pièces

Ce diagramme représente la logique de la génération de mouvement d'une pièce et de la validation du mouvement.



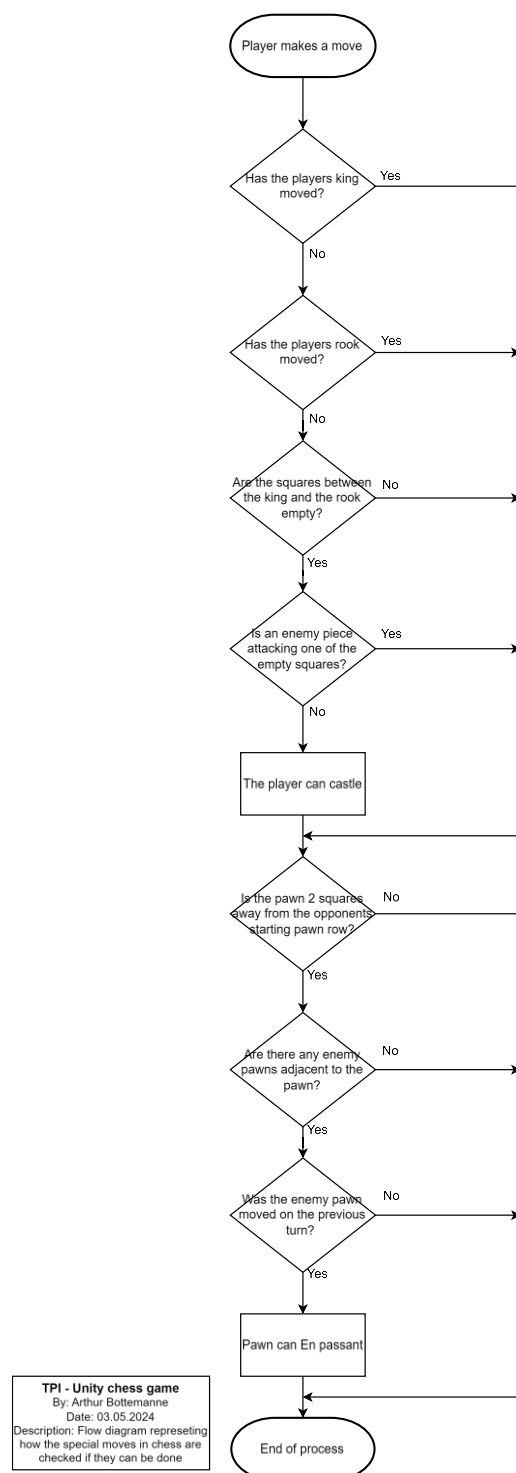


Figure 3: Diagramme de flux de la gestion des mouvements spéciaux

Ce diagramme représente les vérifications des conditions pour le roque et l'en passant aux échecs.

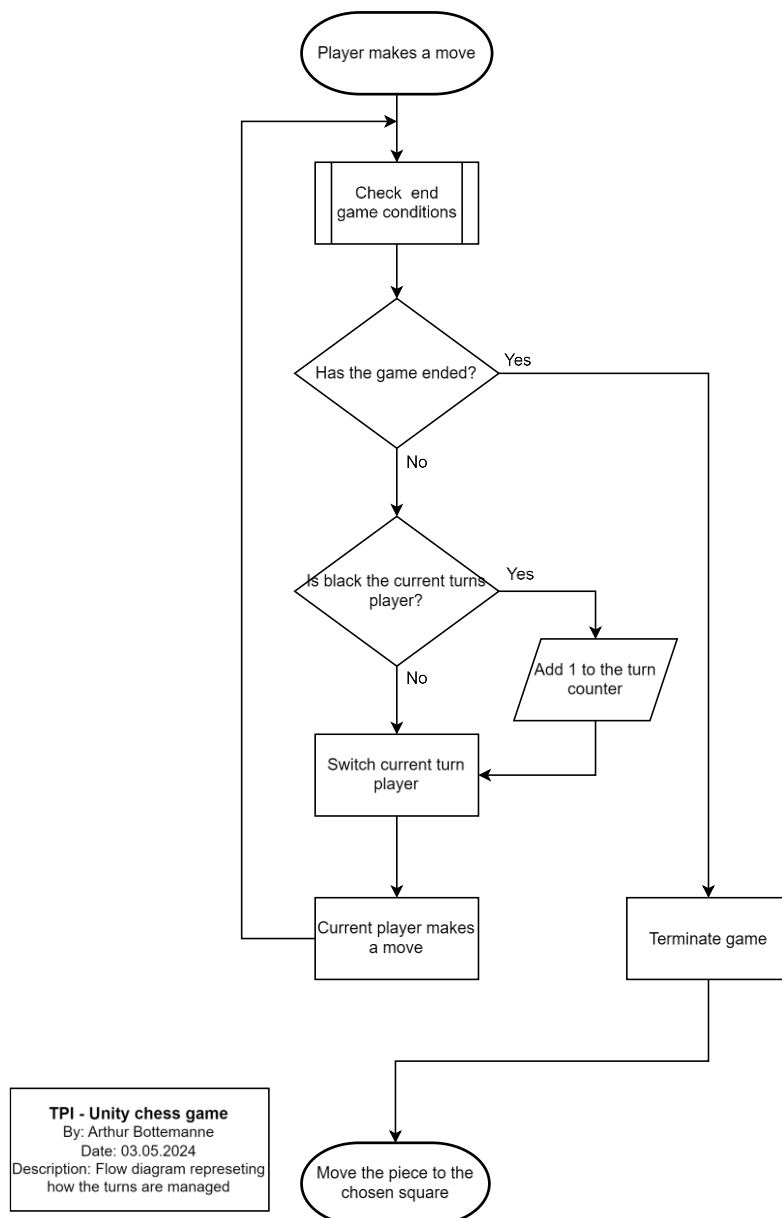


Figure 4: Diagramme de flux de la gestion du tour des joueurs

Ce diagramme représente la logique pour le changement et le compte des tours.

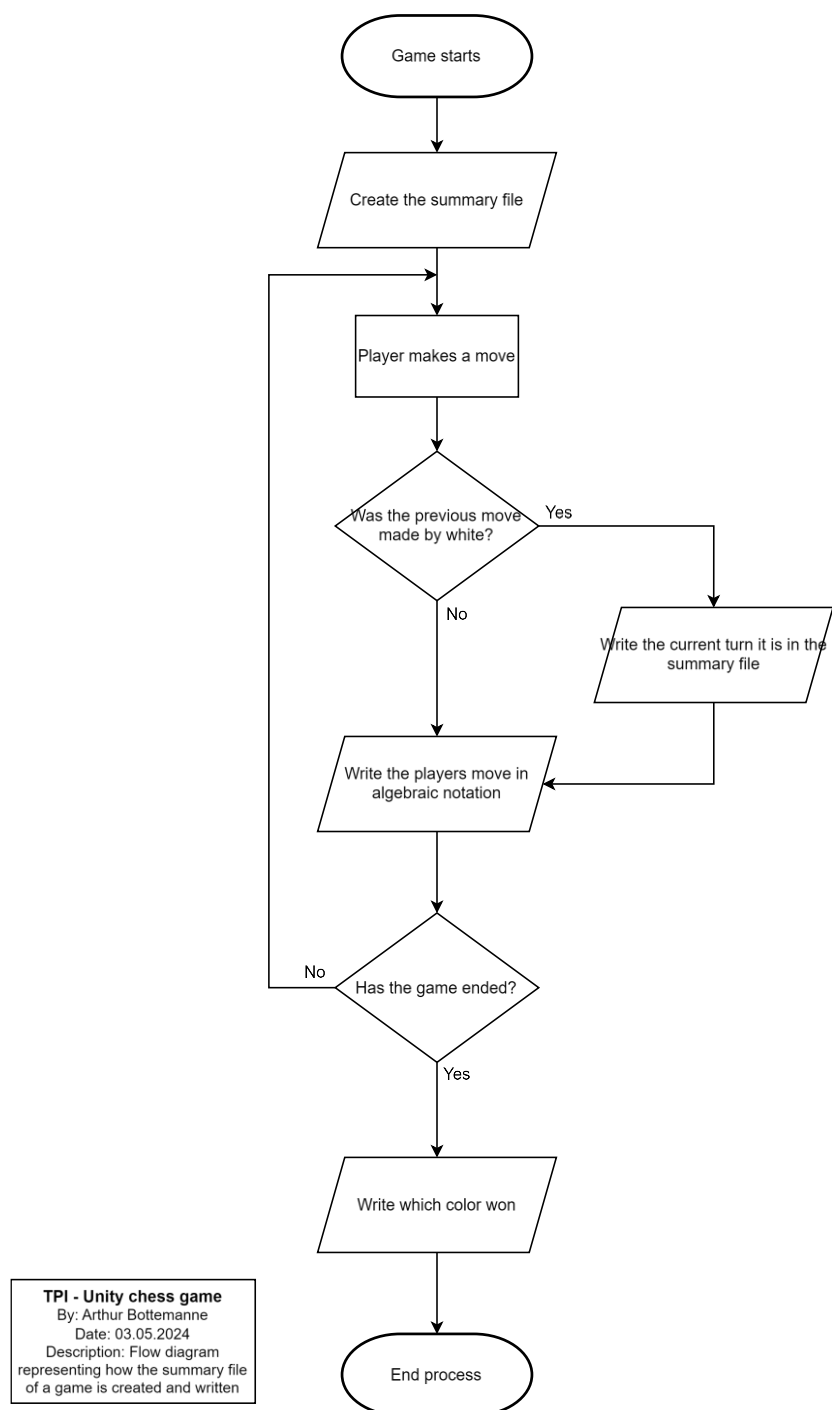


Figure 5: Diagramme de flux de la gestion du fichier récapitulatif

Ce diagramme représente la logique du fichier récapitulatif, quand il écrit les coups et note les tours.

## 2.1.2 Maquettes

Cinq maquettes ont été réaliser pour ce jeu :

- Le menu principal
- La sélection du chronomètre
- La partie en cours
- La proposition d'un match nulle
- Le message de fin de partie

Voici ci-dessous les maquettes :

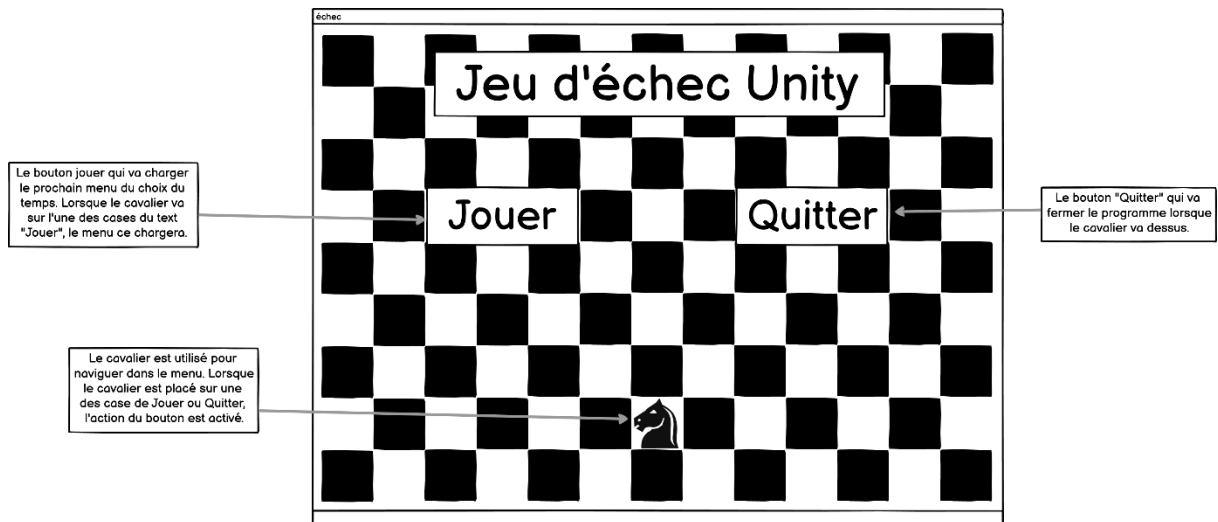


Figure 6: Maquette du menu principal

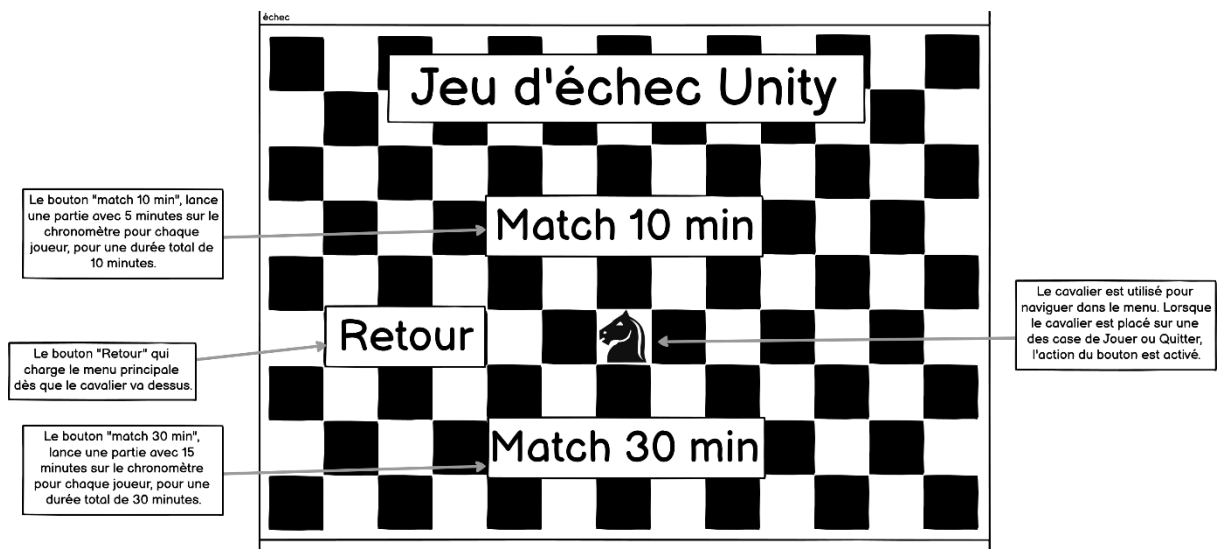


Figure 7: Maquette de la sélection du chronomètre

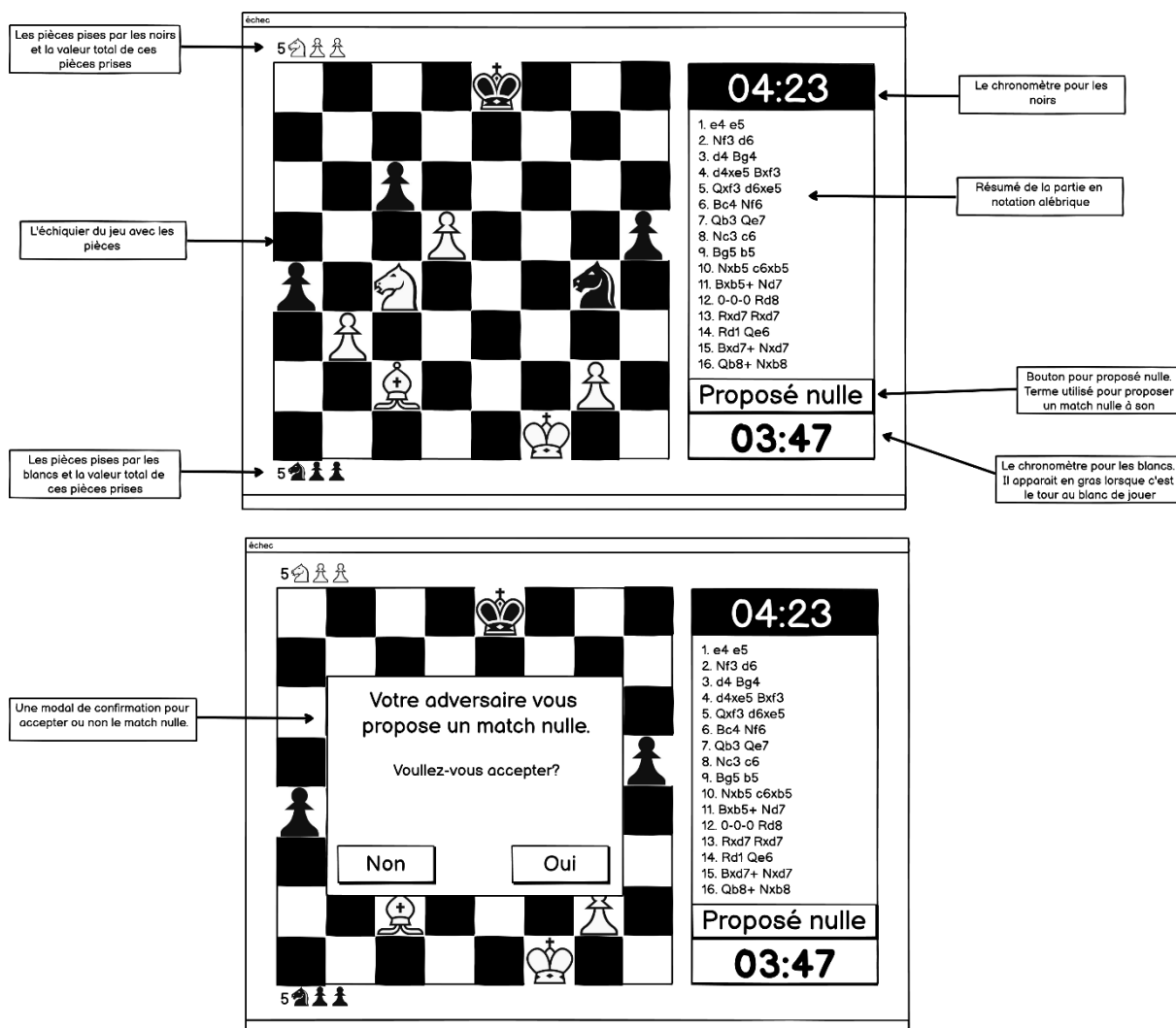


Figure 8: Maquette d'une partie en cours

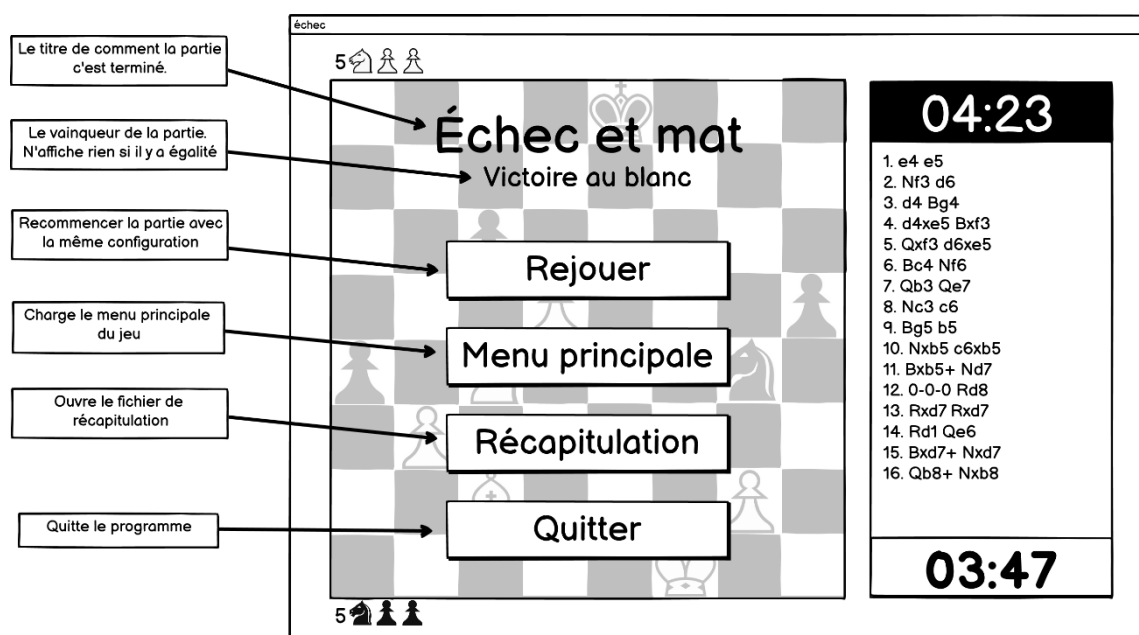


Figure 9: Maquette de la fin d'une partie

## 2.2 Stratégie de test

---

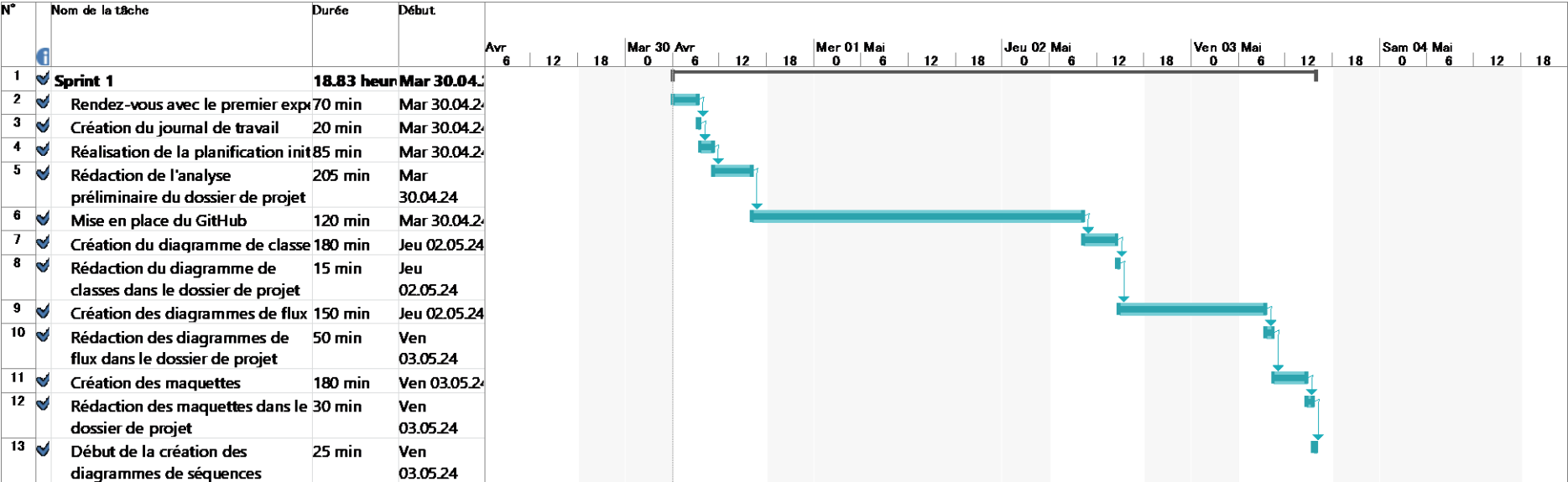
## 2.3 Risques techniques

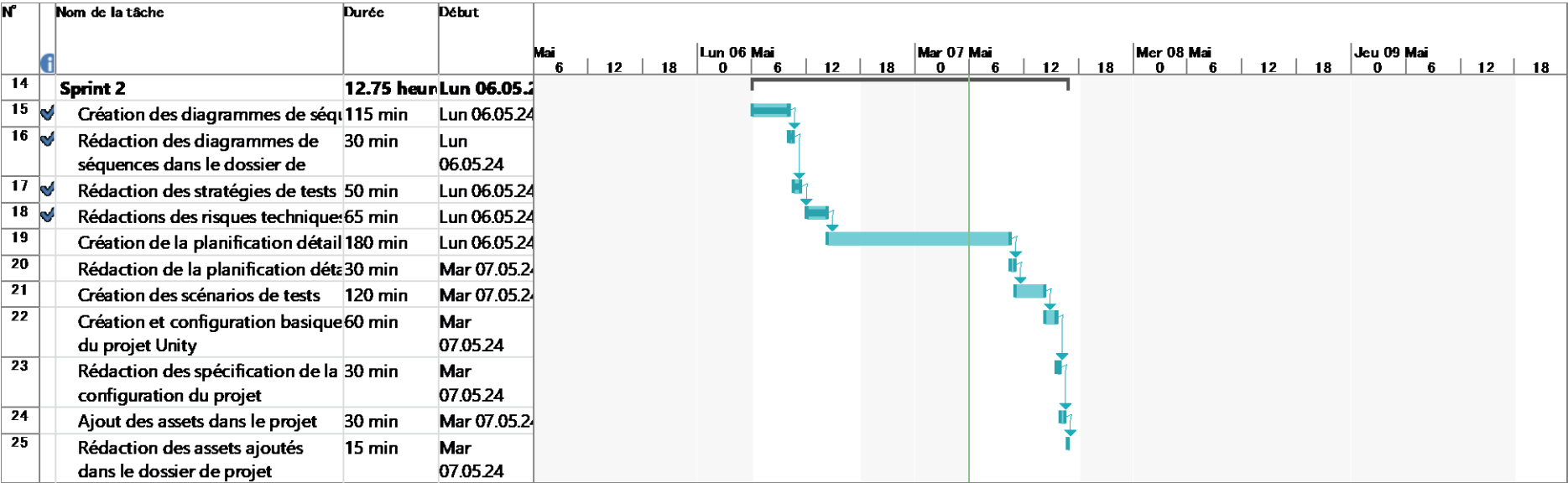
---

Le premier risque majeur serait mes compétences de Unity. Bien que j'aie utilisé Unity quelques fois avant ce projet, il me manque tout de même des connaissances. Pour certaines implémentations que je n'ai pas faites précédemment, cela pourrait se montrer difficile à réaliser. Pour remédier à ce risque, des recherches approfondies au préalable sur Unity seront nécessaire pour bien comprendre comment implémenter la fonction désirée dans Unity, les différentes manières possibles d'implémenter et la meilleure manière de le savoir.

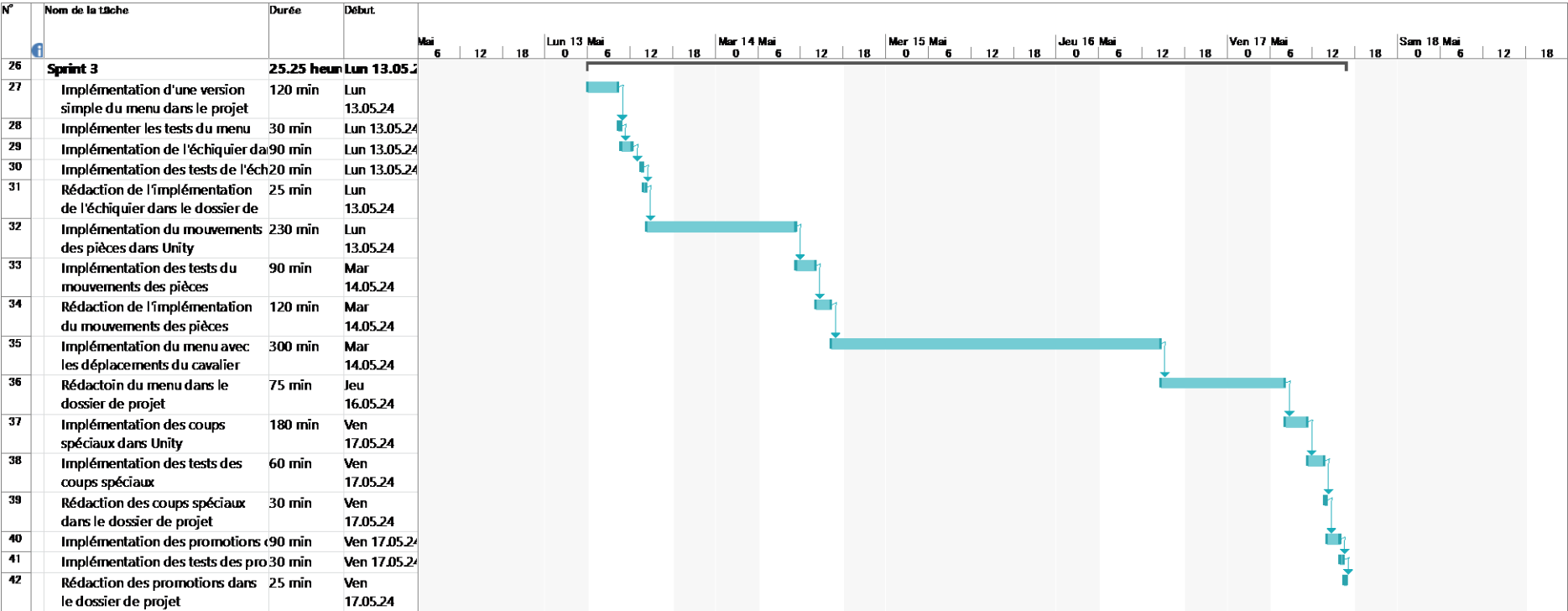
Le deuxième risque que j'anticipe est comment m'assurer que mon programme correspond à 100% avec les règles des échecs. Ceci est compliqué car il y a environ  $10^{40}$  positions légales aux échecs. Dans ces positions, chaque pièce à ses propres règles de déplacement et de capture ce qui peut créer des configurations complexes et avec de nombreuses interactions. Et dans ces configurations, tester et assurer le bon fonctionnement du programme nécessite une analyse minutieuse et exhaustive. Pour éviter ces erreurs, je vais devoir tester le programme avec une variété de positions et de scénarios pour m'assurer qu'il fonctionne correctement dans toutes les situations possibles.

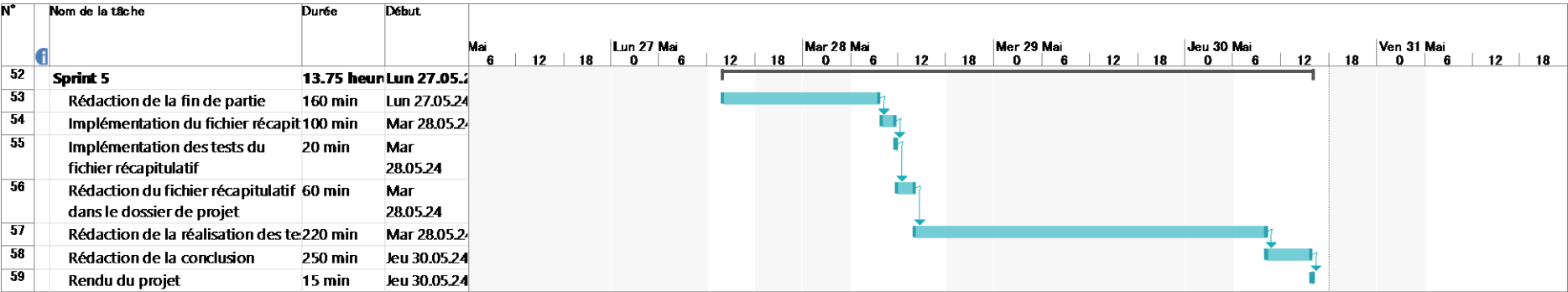
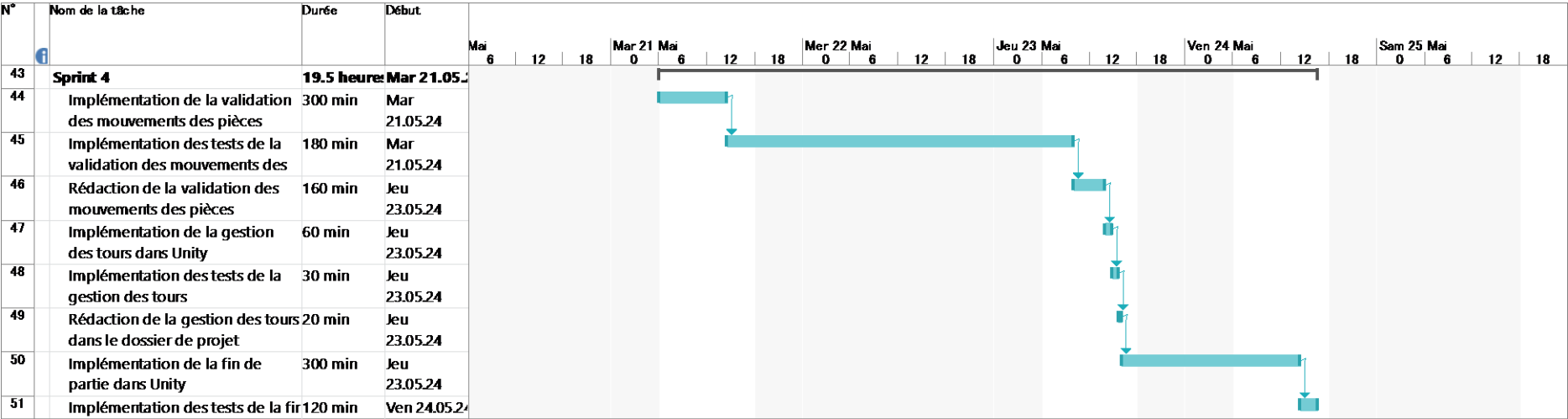
2.4 Planification détaillée













## 2.5 Dossier de conception

### 2.5.1 Diagramme de classes

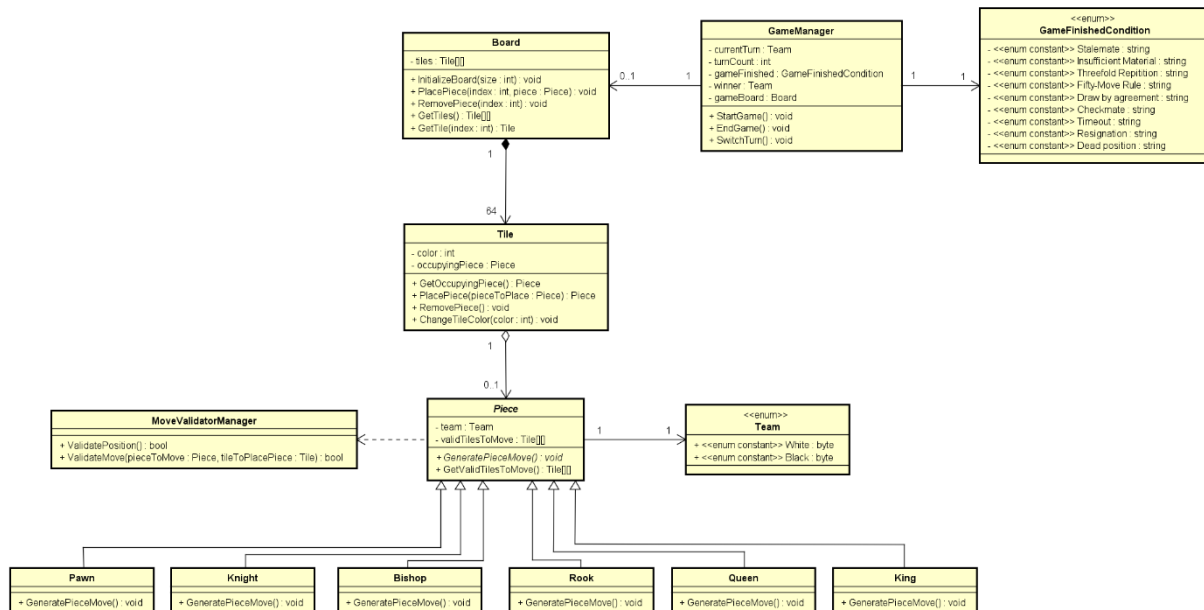


Figure 10: Diagramme de classes

Le diagramme de classes ci-dessus représente la structure des classes que j'envisage d'implémenter. À la fin de la réalisation du projet, un diagramme de classes à jour sera comparé à celui-ci.

Dans ce diagramme, la classe « **GameManager** » gère les fonctionnalités de condition de victoire et de tour, le « **Board** » représente l'échiquier et agit sur les mouvements des pièces et les classes des pièces (« **Pawn** », « **Bishop** » etc...) utilise des concepts de polymorphisme pour implémenter leur propre méthode de mouvement.

## 2.5.2 Diagrammes de séquences

Les diagrammes de séquences réalisés sont concentrés sur le fonctionnement d'une partie, plus spécifiquement sur l'initialisation d'une partie, du mouvement d'une pièce et de la fin d'une partie.

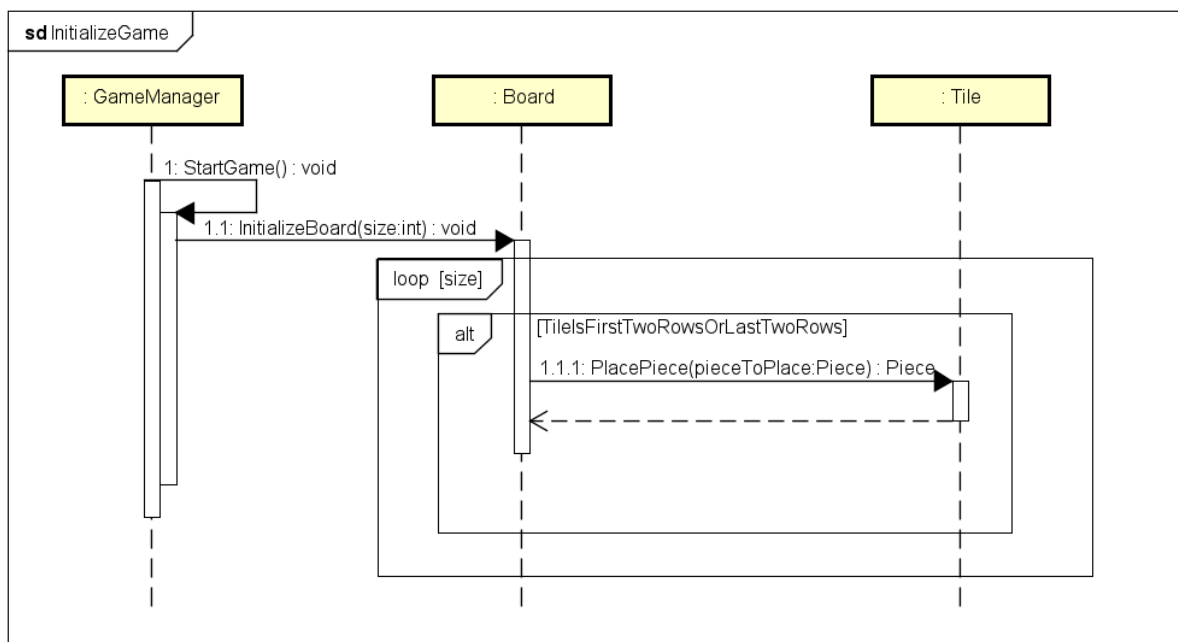


Figure 11: Diagramme de séquence de l'initialisation d'une partie

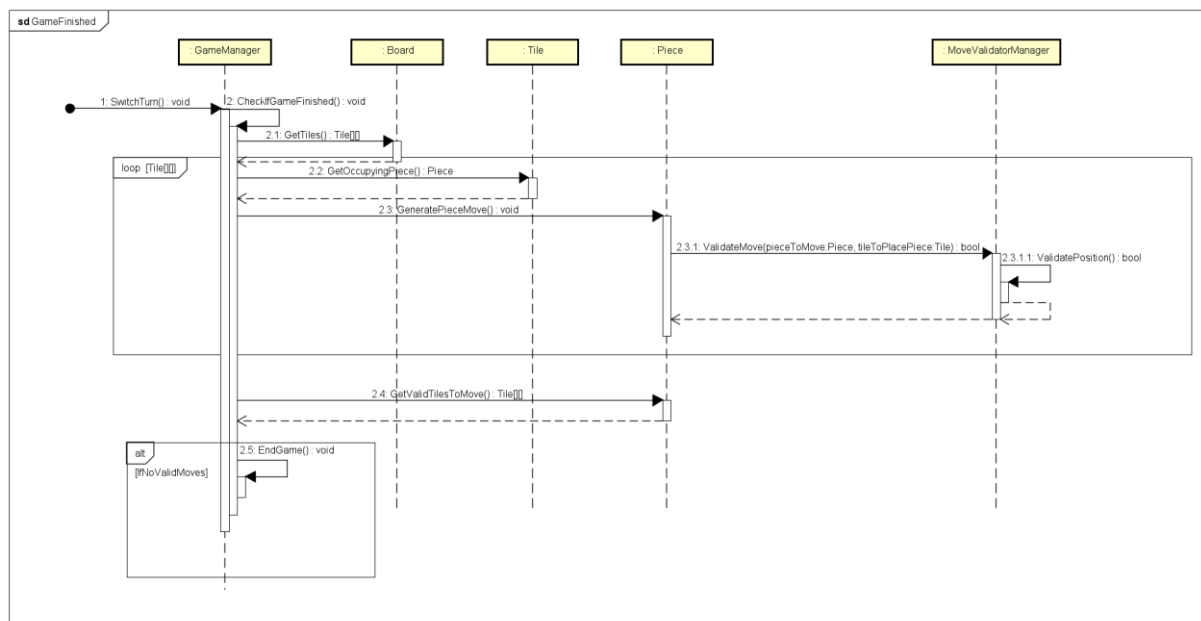


Figure 7: Diagramme de séquence de la fin d'une partie

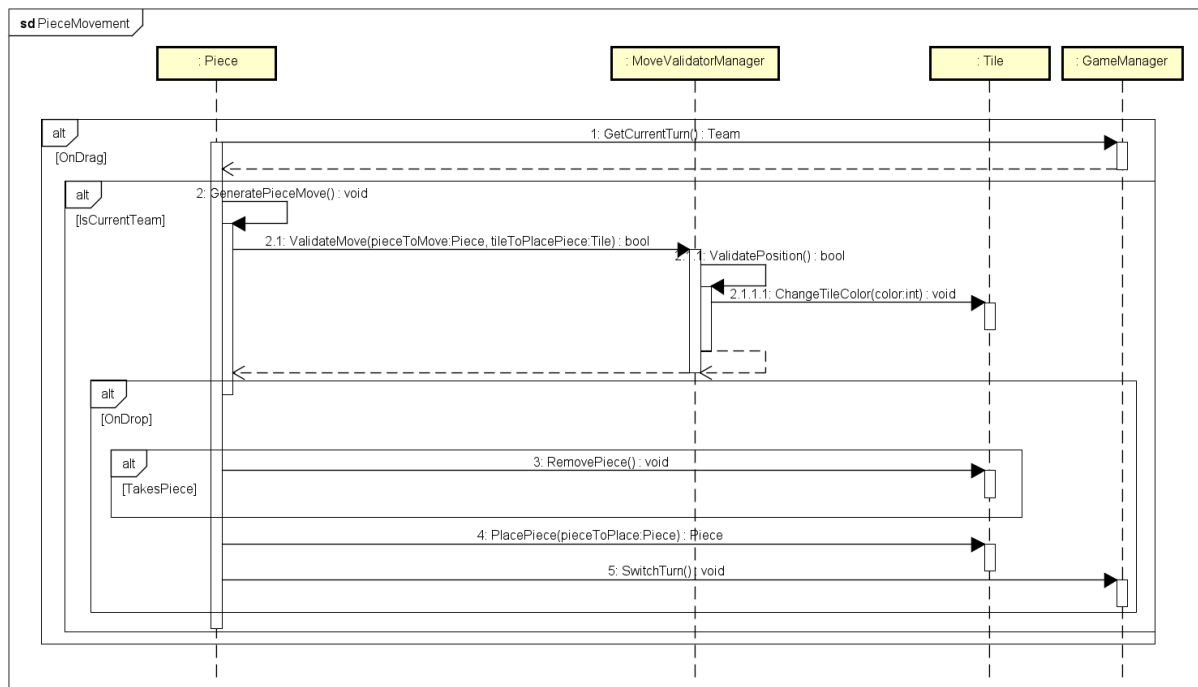


Figure 6: Diagramme de séquence du mouvement des pièces

Fournir tous les document de conception:

*le choix du matériel HW*

*le choix des systèmes d'exploitation pour la réalisation et l'utilisation*

*le choix des outils logiciels pour la réalisation et l'utilisation*

*site web: réaliser les maquettes avec un logiciel, décrire toutes les animations sur papier, définir les mots-clés, choisir une formule d'hébergement, définir la méthode de mise à jour, ...*

*bases de données: décrire le modèle relationnel, le contenu détaillé des tables (caractéristiques de chaque champs) et les requêtes.*

*programmation et scripts: organigramme, architecture du programme, découpage modulaire, entrées-sorties des modules, pseudo-code / structogramme...*

***Le dossier de conception devrait permettre de sous-traiter la réalisation du projet !***

## 3 Réalisation

### 3.1 Dossier de réalisation

*Décrire la réalisation "physique" de votre projet*

- *les répertoires où le logiciel est installé*
- *la liste de tous les fichiers et une rapide description de leur contenu (des noms qui parlent !)*
- *les versions des systèmes d'exploitation et des outils logiciels*
- *la description exacte du matériel*
- *le numéro de version de votre produit !*
- *programmation et scripts: bibliothèques externes, dictionnaire des données, reconstruction du logiciel - cible à partir des sources.*

*NOTE : Évitez d'inclure les listings des sources, à moins que vous ne désiriez en expliquer une partie vous paraissant importante. Dans ce cas n'incluez que cette partie...*

### 3.2 Description des tests effectués

*Pour chaque partie testée de votre projet, il faut décrire:*

- *les conditions exactes de chaque test*
- *les preuves de test (papier ou fichier)*

- *tests sans preuve: fournir au moins une description*

### 3.3 Erreurs restantes

---

*S'il reste encore des erreurs:*

- *Description détaillée*
- *Conséquences sur l'utilisation du produit*
- *Actions envisagées ou possibles*

### 3.4 Liste des documents fournis

---

*Lister les documents fournis au client avec votre produit, en indiquant les numéros de versions*

- *le rapport de projet*
- *le manuel d'Installation (en annexe)*
- *le manuel d'Utilisation avec des exemples graphiques (en annexe)*
- *autres...*

## 4 Conclusions

*Développez en tous cas les points suivants:*

- *Objectifs atteints / non-atteints*
- *Points positifs / négatifs*
- *Difficultés particulières*
- *Suites possibles pour le projet (évolutions & améliorations)*



---

## 5 Annexes

### 5.1 Résumé du rapport du TPI / version succincte de la documentation

---

### 5.2 Sources – Bibliographie

---

*Liste des livres utilisés (Titre, auteur, date), des sites Internet (URL) consultés, des articles (Revue, date, titre, auteur)... Et de toutes les aides externes (noms)*

### 5.3 Journal de travail

---

Date	Durée	Activité	Remarques

### 5.4 Manuel d'Installation

---

### 5.5 Manuel d'Utilisation

---

### 5.6 Archives du projet

---

*Media, ... dans une fourre en plastique*