École Publique d'Ingénieurs en 3 ans

Génie logiciel et patrons de conception

2023-2024

MISE EN PLACE DU PROJET

Guillain Le Goff
guillain.le-goff@ecole.ensicaen.fr
Léo Bretagne
leo.bretagne@ecole.ensicaen.fr
Lucas Ollier
lucas.ollier@ecole.ensicaen.fr
Clément Cartro
clement.cartro@ecole.ensicaen.fr
Moris Lorys Kamgang
moris-lorys.kamgang@ecole.ensicaen.fr
Hamid Ben-Omar
hamid.ben-omar@ecole.ensicaen.fr
Achraf Aboulakjam
achraf.aboulakjam@ecole.ensicaen.fr

ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

www.ensicaen.fr

1. Diagramme des cas d'utilisation

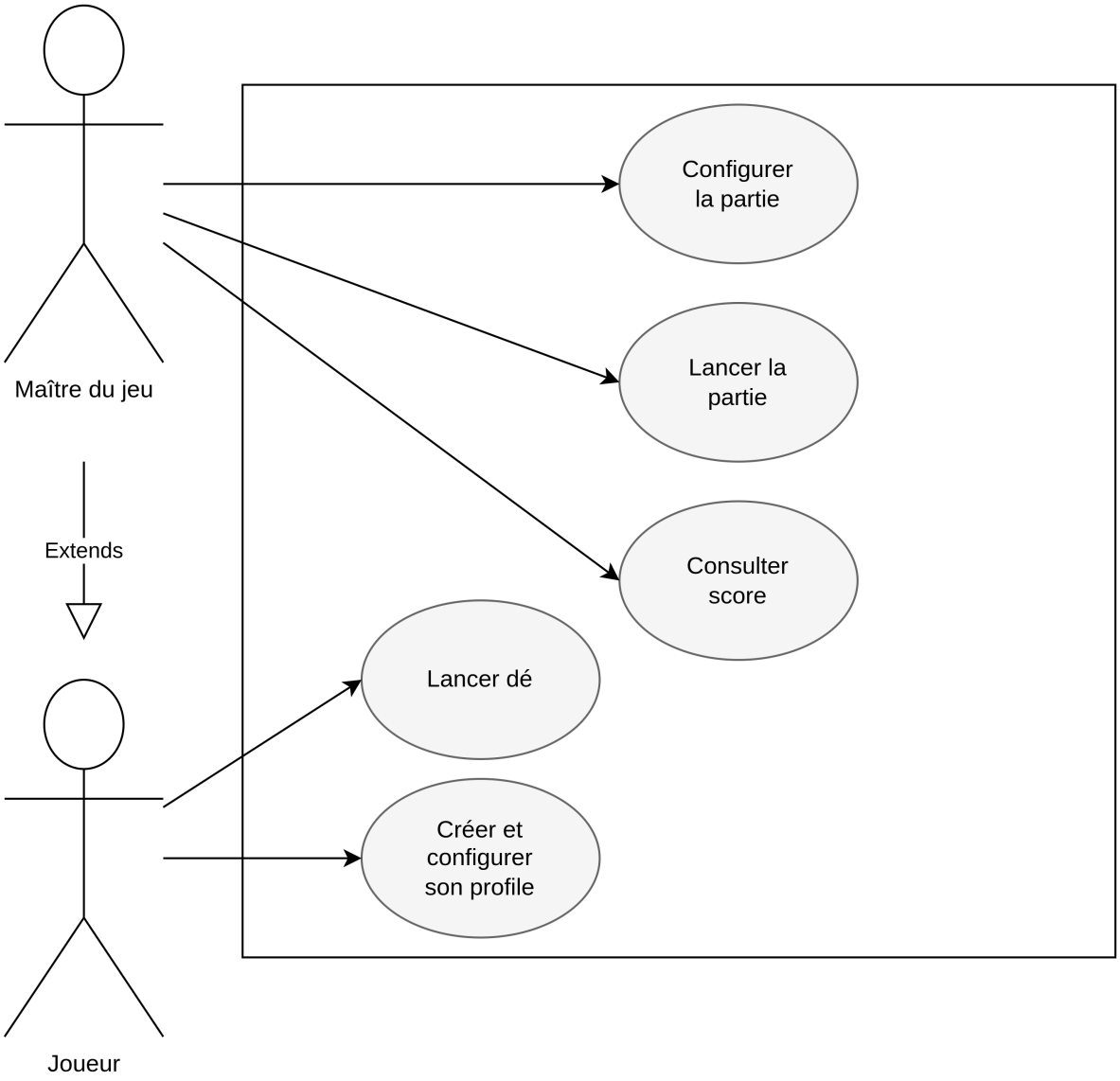

Figure 1. *Diagramme des cas d'utilisation*

2. Analyse des risques du projet

Nous divisons les risques en 3 catégories :

- Les risques techniques
- Les risques organisationnels
- Les risques fonctionnels

Risques du projet :

Risques du projet	Gravité	Probabilité	Criticité
Manque de connaissances techniques : Java, JavaFX, Git, JSON, XML,	Majeure	Modérée	Élevée
Différents environnements de développement (Windows, Linux, Mac)	Significatif	Certaine	Élevée
Liste des tâches pas assez exhaustive	Significatif	Modérée	Moyen
Répartition inégales des tâches	Significatif	Probable	Moyen
Difficultés à estimer la durée des tâches	Majeure	Très probable	Très élevée
Communication insufisante entre les membres de l'équipe	Sévère	Modérée	Très élevée
Indisponibilité d'un membre	Significatif	Improbable	Moyen
Fonctionnalités trop complexes à mettre en œuvre	Majeure	Modérée	Élevée
Fonctionnalités défectueuses	Sévère	Probable	Très élevée
Produire du code incompatible avec le code des autres membres	Sévère	Probable	Très élevée

Figure 2. *Risques*

Solutions à ces risques :

Risques du projet	Solutions
Manque de connaissances techniques : Java, JavaFX, Git, JSON, XM	Se former à ces technologies en dehors des séances, demander de l'aide aux membres
Différents environnements de développement (Windows, Linux, Mac)	Discuter de son environnement avec le responsable de version pour vérifier la compatibilité
Liste des tâches pas assez exhaustive	Prendre le temps de bien définir les objectifs du projet et son fonctionnement
Répartition inégales des tâches	Réviser et ajuster régulièrement la répartition des tâches
Difficultés à estimer la durée des tâches	S'appuyer sur ses expériences en développement, diviser les tâches en sous tâches
Communication insufisante entre les membres de l'équipe	Utilisation d'un groupe de messagerie instantannée pour communiquer, organiser des bilans réguliers et des pauses pour discuter
Indisponibilité d'un membre	Déléguer le travail à réaliser aux membres ayant le moins de travail à priori, Prévenir le reste de l'équipe si possible
Fonctionnalités trop complexes à mettre en œuvre	Bien évaluer la difficulté de la fonctionnalité en fonction du temps disponible, Simplifier la fonctionnalité, la découper en plusieurs sous tâches faciles
Fonctionnalités défectueuses	Mettre en place des tests rigoureux pour détecter les problèmes, Effectuer du peer reviewing pour identifier les erreurs potentielles.
Produire du code incompatible avec le code des autres membres	Établir des normes de codage claires et demander des conseils/consignes au responsable de version

Figure 3. *Solutions*

3. Diagramme de classes métier

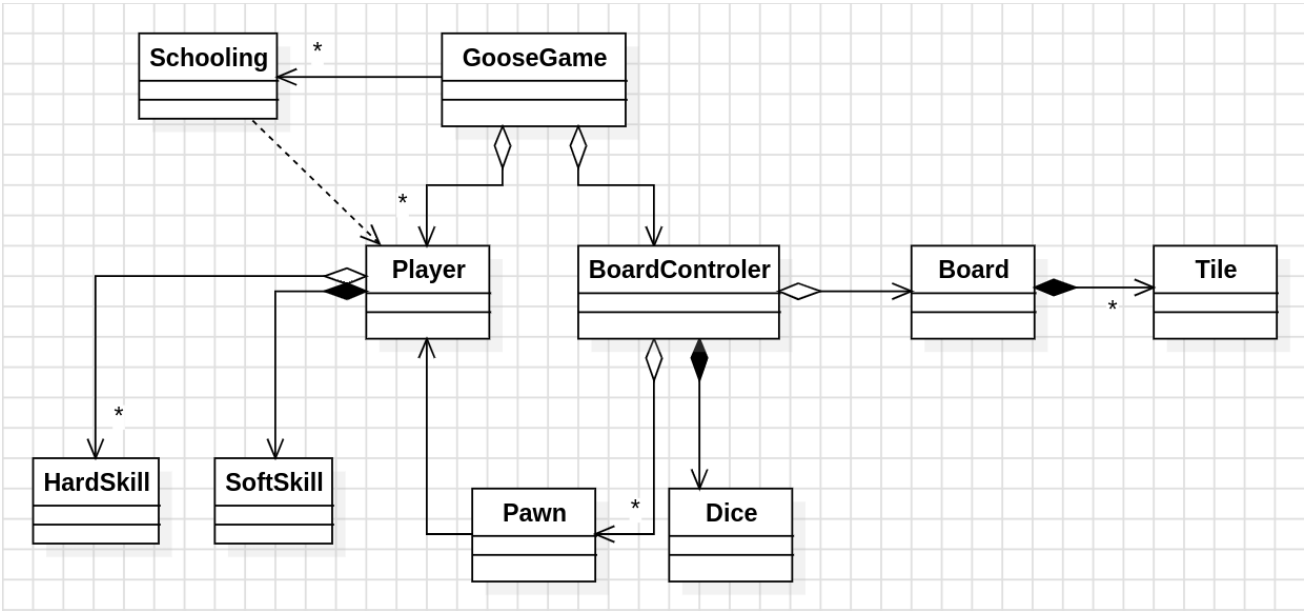

Figure 4. *Diagramme des classes métier*

4. Liste des tâches 1er MVP

4.1. Limitations

- Un plateau avec un seul type de case sans effet.
- Parties avec un seul joueur.
- 1 seul HardSkill sans effet.
- 1 seul SoftSkill sans effet
- Plateau en ligne droite, écrit dans une classe qui implémente ce qui devrait être le reader.
- 1 seule langue et le minimum de texte pour le moment.

4.2. Objectifs

- Possibilité de démarrer la partie.
- Création du personnage (donc déterministe pour le moment).
- Possibilité de voir le plateau.
- Possibilité de lancer le dé et voir le résultat à chaque tour.
- Possibilité de voir les déplacements du pion (une téléportation vers la nouvelle case).
- La partie se termine lorsque le joueur a terminé.

4.3. Aperçu des tâches à faire pour le 1er MVP

TILES

- Ecrire une classe Case
- Ecrire une interface TileType
- Ecrire une classe WeekEnd qui implémente TileType

BOARD_READER

- Ecrire une interface BoardReader
- Ecrire une classe qui implémente BoardReader

HARDSKILL

- Ecrire une interface HardSkill
- Ecrire une classe NoEffectHardSkill qui l'implémente

SOFTSKILL

- Ecrire une interface SoftSkill avec une méthode
- Ecrire une classe Assidu qui implémente SoftSkill

PLAYER

- Monteur pour la création

SCHOOLING

- Ecrire une interface Schooling
- Ecrire une classe NoTraining implémente Schooling

LANGUAGE

- Ecrire une interface Language
- Ecrire une interface LanguageFR qui implémente Language

BOARD

- Réaliser l'interface graphique (FXML)
- Afficher le plateau (les cases)
- Afficher les Pawn

PAWN

- Méthode move() qui change sa position.

BOARDCONTROLER

- Méthode play()
- Lancer un dé
- Afficher le dé (responsabilité unique ?)
- Fin de la partie si le joueur dépasse la ligne

DICE

- Se lancer

GOOSEGAME

- Accéder à SCHOOLING pour créer un joueur
- Démarrer la partie avec 1 joueur