Étude préalable:
Simulation de poursuite-évasion compétitive entre agents informatiques intelligents

Tuteur: Guénaël Cabanes

Matias Amaglio
Maëlle Bitsindou
Luc Dechezlepretre
Célie Ponroy
Année universitaire 2024- 2025

Table des matières

1. Présentation du projet

Le client principal de notre projet est le tuteur de notre projet, Monsieur Cabanes. Néanmoins, les cibles qui pourront utiliser le produit final seront à la fois des personnes qui souhaitent découvrir et étudier certains comportements d'IA, plus précisément les algorithmes utilisés, mais également des personnes qui souhaitent juste jouer contre l'une des IA.

Pour la réalisation optimale de ce projet, l'application devra répondre à deux besoins principaux. En premier lieu, notre projet sera à but éducatif et de recherche, pour permettre aux personnes expérimentées ou non de comprendre les différents comportements des IA. En deuxième lieu, notre application devra également répondre à des besoins ludiques pour permettre à l'utilisateur de jouer à la simulation. L'utilisateur sera capable de choisir un niveau de difficulté, différentes cartes… Lors de la première itération, il pourra seulement prendre le rôle du prisonnier, mais par la suite il pourra jouer les 2 rôles. Pour répondre aux besoins de façon à avoir un produit réussi, nous devons implémenter trois fonctionnalités critiques :

- Implémentation des comportements des agents intelligents, ce qui va être la base de notre projet,
- L'outil d'analyse des résultats, qui permettra de visualiser la carte choisie avec les probabilités de chaque agent,
- Création de différentes vues, pour permettre de voir l'évolution de la simulation.

Par rapport aux produits déjà présents sur le marché, notre application servira de complément à d'autres projets qui existent déjà car il est à la fois à but éducatif et ludique.

2. Liste des fonctionnalités du système

Pour obtenir un produit qualitatif, nous devrons implémenter différentes fonctionnalités en fonction du modes de simulation sélectionné:

- Choisir un mode, intéractif ou non
- Dans le mode **interactif**,
  - ➢ Choisir l'agent incarné (prisonnier ou gardien)
    - → L'utilisateur pourra accéder à un menu pour sélectionner l'agent qu'il souhaitera incarner.
  - ➢ Choisir le niveau des IA
    - → L'utilisateur pourra choisir le niveau de l'IA (faible, moyen et avancé)
  - ➢ Jouer à la simulation
    - → L'utilisateur pourra jouer un des rôles et essayer de battre l'agent.
  - ➢ Consulter et analyser les résultats de la partie grâce à l'historique des déplacements
    - → L'utilisateur pourra analyser chaque probabilité de déplacement sur l'agent en question.
- Dans le mode **non interactif**,
  - ➢ Choisir le niveau des IA
    - → L'utilisateur pourra choisir le niveau des IA (faible, moyen et avancé)
  - ➢ Regarder un déroulement de la simulation
    - → L'utilisateur va pouvoir analyser chaque étape de la simulation et visualiser les probabilités sur la position de l'adversaire.

3. Analyse des cas d'utilisation

- **Cas d'utilisation**

Nous avons tout d'abord listé tous les cas d'utilisation et les acteurs de notre application.

Acteurs	- L'utilisateur (personne qui utilise l'application)
Cas d'utilisation	- Choisir une partie en mode interactif - Choisir une partie en mode non interactif - Prendre le contrôle d'un des agents (prisonnier ou gardien) - Choisir le niveau des IA - Regarder le déroulement des parties - Consulter un bilan de la partie

- **Diagramme de cas d'utilisation**

L'application n'ayant pas besoin de services extérieurs et n'ayant pas besoin de rôle spécifique pour l'administration ou la maintenance, il n'y aura qu'un seul utilisateur à la fois. Il peut choisir le mode interactif ou non interactif, prendre le contrôle d'un personnage, choisir le niveau des agents, regarder le déroulé d'une partie et consulter un bilan de partie s'il est en mode interactif.

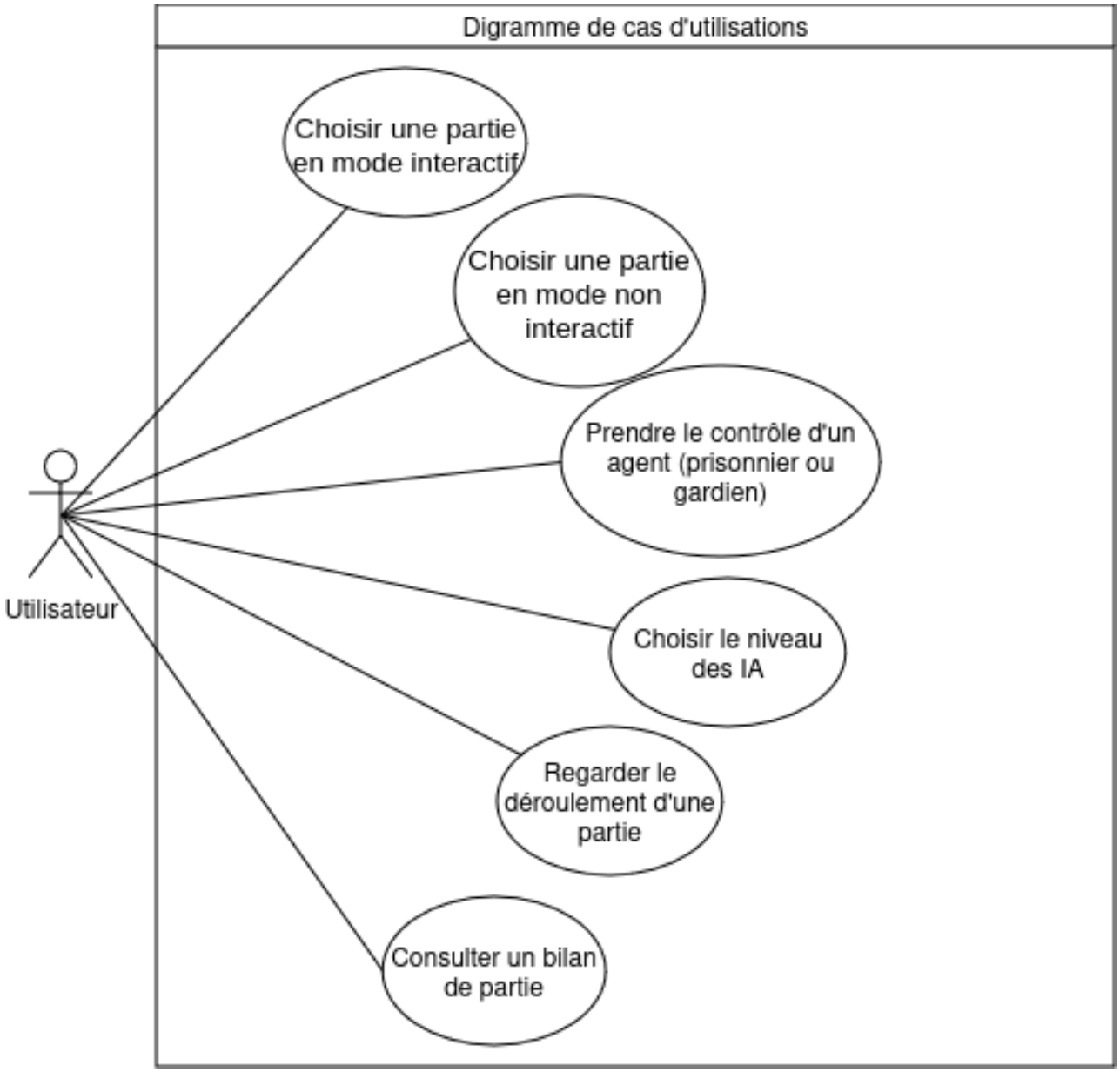


Diagramme de cas d'utilisation de l'application finale

- **Scénarios**

Nous avons établi 3 scénarios pour mieux visualiser chaque cas d'utilisation de l'application.

➢ **Scénario 1 :**

→ L'utilisateur lance l'application.

→ L'utilisateur choisit de jouer une partie.

→ Il choisit de prendre le rôle du prisonnier.

→ Il choisit un niveau de difficulté.

→ L'utilisateur joue sa partie et se fait attraper par l'agent qui incarne le gardien.

→ Il prend la décision d'analyser la partie qu'il vient de jouer.

→ Le système affiche à la suite chaque coups de la partie.

→ L'utilisateur quitte l'application.

➢ **Scénario 2 :**

→ L'utilisateur lance l'application.

→ L'utilisateur choisit de jouer une partie.

→ Il choisit de prendre le rôle du gardien.

→ Il choisit un niveau de difficulté.

→ L'utilisateur joue un certain moment.

→ L'utilisateur joue sa partie et attrape l'agent qui incarne le prisonnier.

→ L'utilisateur quitte l'application.

➢ **Scénario 3 :**

→ L'utilisateur lance l'application.

→ L'utilisateur choisit de jouer une partie.

→ Il choisit de prendre le rôle du prisonnier.

→ Il choisit un niveau de difficulté.

→ L'utilisateur joue sa partie et se fait attraper par l'agent qui incarne le gardien.

→ L'utilisateur sélectionne l'option recommencer la partie.

- **Conditions de validation**

Notre projet va être un projet pouvant avoir différentes versions. Notre objectif premier est d'avoir une version stable et fonctionnelle pour ensuite avoir une version plus avancée. Voici donc les différentes évolutions que pourra avoir notre projet.

➢ Version minimale du produit:
  - Prendre le contrôle d'un des agents
  - Accéder à la vue de l'historique
  - Accéder au mode non interactif de la simulation

➢ Version médium du produit:
  - Obstacles
  - Avoir des niveaux de difficultés
  - Accéder au mode intéractif de la simulation

➢ Version avancée du produit:
  - Temps réel
  - Des cartes aléatoires
  - Point de départ aléatoire
  - Avoir plusieurs IA (à voir selon le temps d'apprentissage)

4. Modélisation du système

- **Le diagramme d'activités de l'utilisateur**

Voici le diagramme d'activité de l'utilisateur, il représente tout le processus de l'application en commençant par le lancement de l'application jusqu'à la fin.

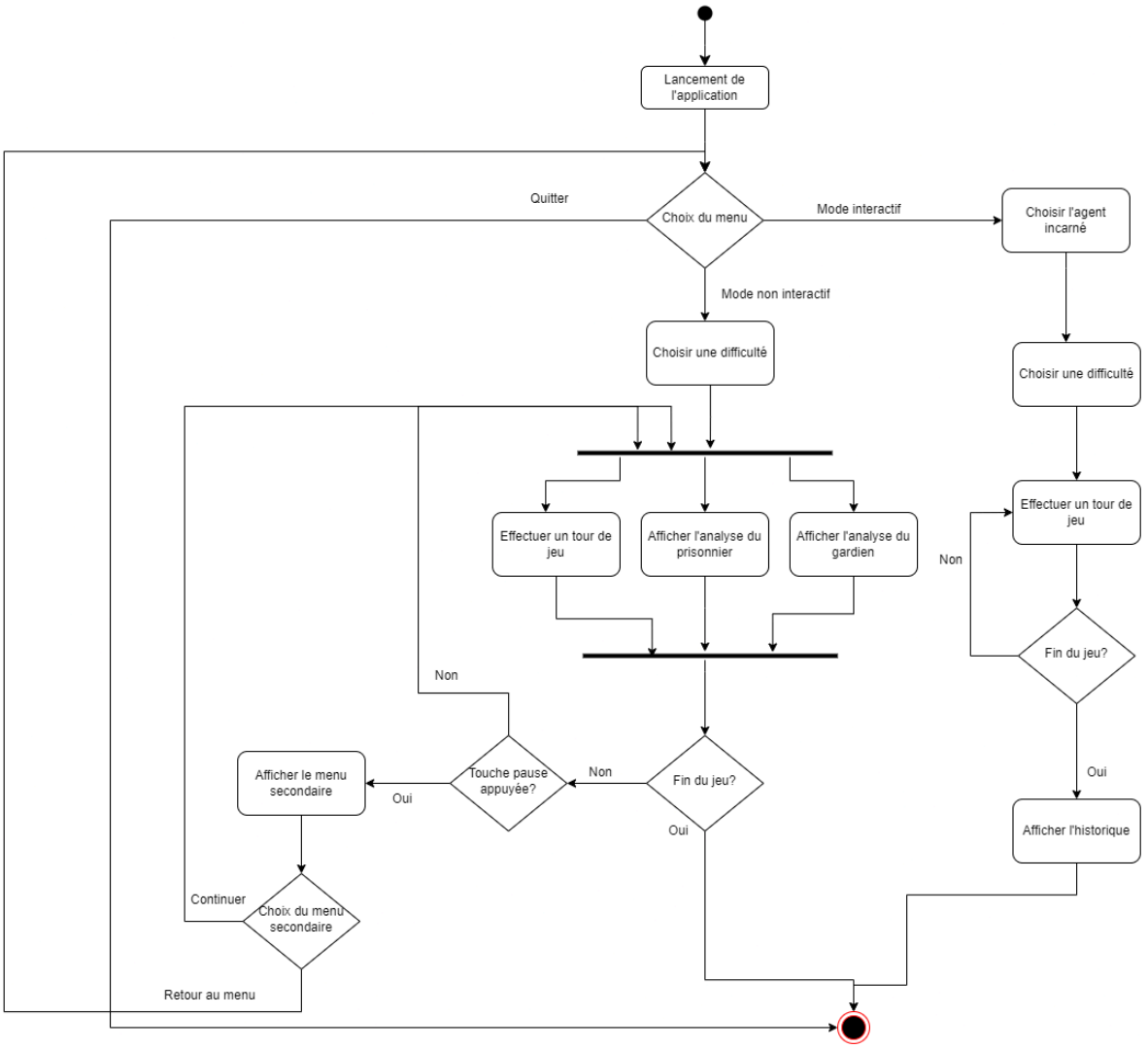


Diagramme de d'activité de l'utilisateur

- **Le diagramme d’activités de l’apprentissage par renforcement**

Ce diagramme d’activité représente l’apprentissage par renforcement que nous allons utiliser dans notre projet. L'algorithme reçoit un historique de partie, lui attribue une note, positive ou négative en prenant en compte nos critères de notation. En fonction de cette note, les poids du réseau de neurones sont modifiés. L’algorithme vérifie ensuite si son nombre d’itération a été atteint, si oui il s'arrête sinon il boucle et reçoit les nouveaux historiques.

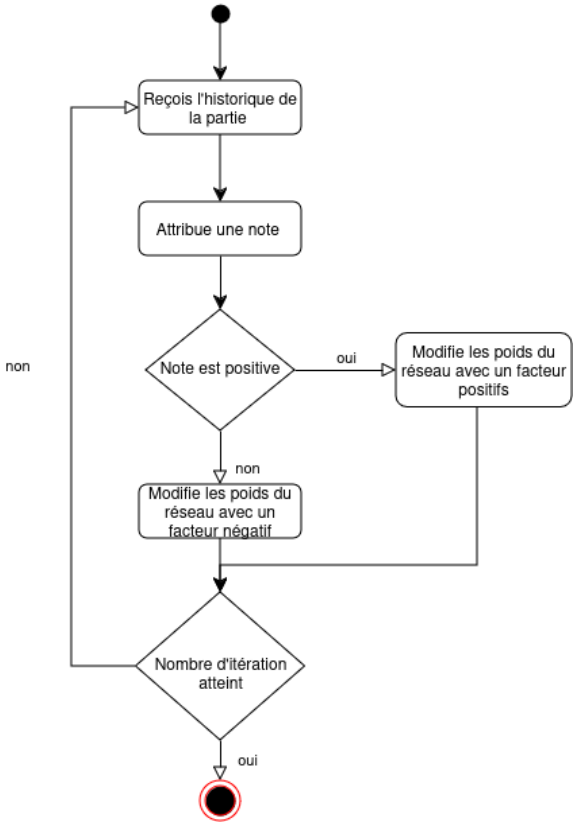


Diagramme d’activités de l’apprentissage par renforcement

5. Maquettes de l'application

Pour un meilleur visuel de l'application, nous avons défini des sprites pour chaque rôle, ainsi que des maquettes.

- **Sprites**

Pour symboliser les différents agents nous avons choisi des sprites pré-dessinés pour les rôles de gardien et de prisonnier.

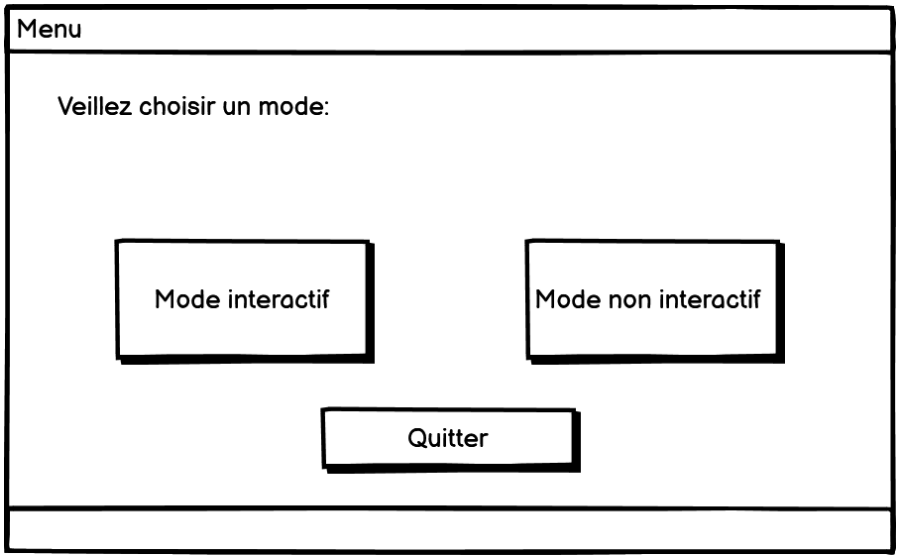
Sprite du gardien

Sprite du prisonnier

Ces sprites permettront à la fois d'avoir un produit final, propre et "réaliste" mais permettra aussi de s'adapter dans les cas où il y aurait plusieurs rôles, en changeant par exemple les couleurs des uniformes. Dans la suite de notre projet, nous pourrons également ajouter des sprites pour les différents pièges et obstacles.

- **Maquettes**

Pour une meilleure visualisation de l'apparence de l'application finale, nous avons créé les maquettes de chacune des vues.



Menu principal de l'application - Choix du mode

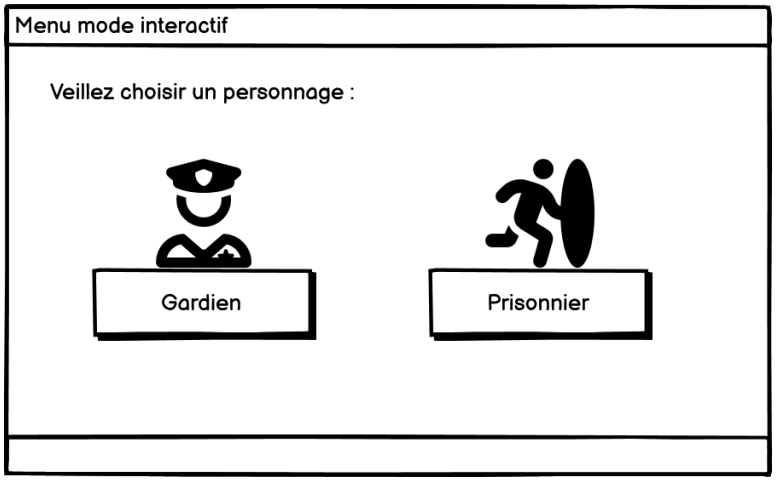
Cette maquette représente le menu principal. Ce menu sera le premier à s'afficher lors du lancement de l'application. Elle permet à l'utilisateur de sélectionner un des modes (intéractif ou non) pour la simulation.

Après cette vue, en fonction de son choix, plusieurs possibilités s'offrent à l'utilisateur. Soit la simulation se lancera en mode non interactif, comme ci-dessous

Vue de l'application - Mode de la simulation

En choisissant ce mode, l'utilisateur pourra suivre les déplacements et le nombre d'itérations des agents dans la fenêtre "Simulation". Dans les fenêtres d'analyse, il pourra voir les probabilités de présence des adversaires. L'analyse du gardien affichera les probabilités de présence du prisonnier et inversement.

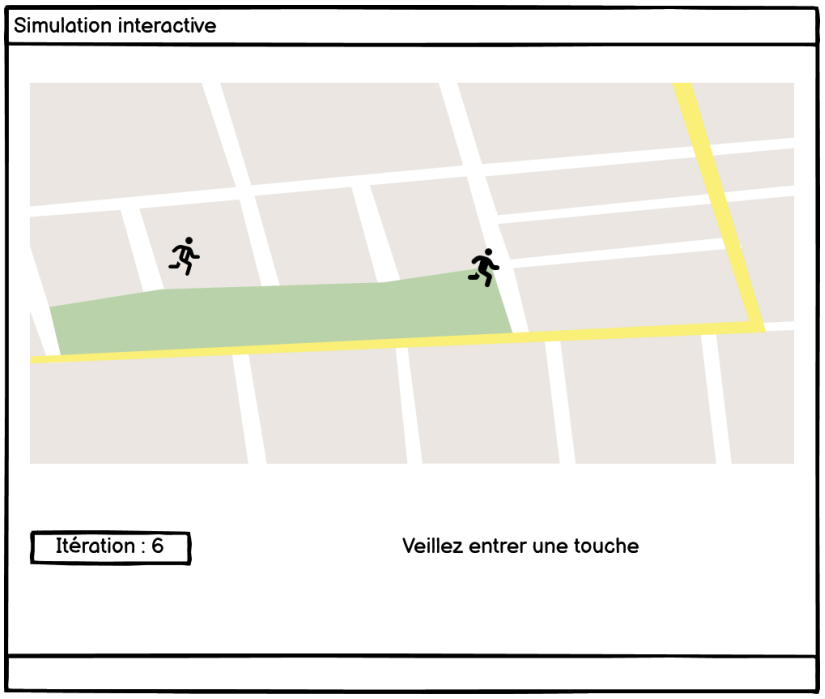
Soit un autre menu s'ouvrira dans le cas du mode interactif, pour choisir l'un des rôles que l'utilisateur souhaite incarner.



Menu du choix du personnage joué - Mode interactif

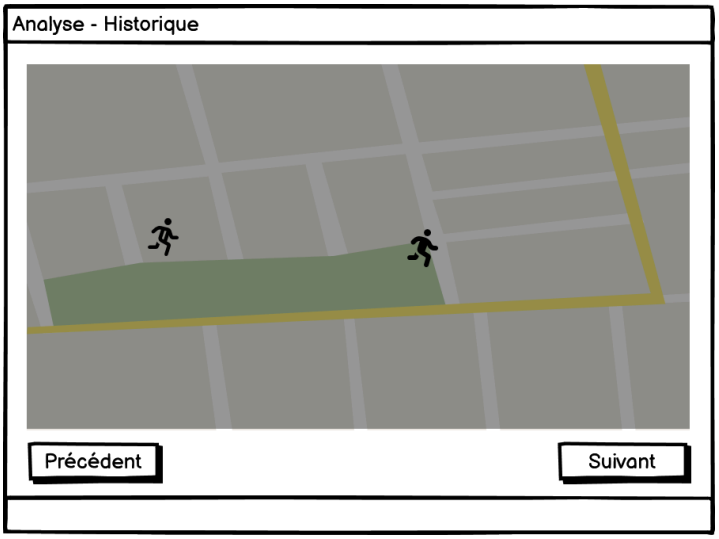
Après avoir sélectionné le personnage, la partie peut commencer, la fenêtre de la simulation interactive va donc s'afficher.

Sur cette maquette, l'utilisateur va voir le nombre d'itérations ainsi que ses déplacements et ceux de l'agent concurrent. Tous les déplacements se feront grâce à des touches du clavier.



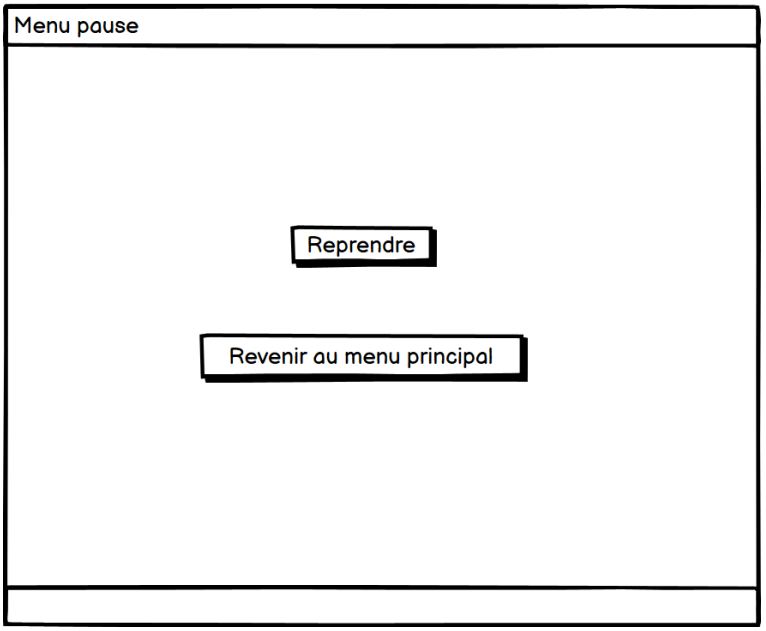
Vue de l'application - Mode de la simulation interactive

Après la fin de la partie, l'utilisateur va pouvoir accéder à l'historique des déplacements et aux différentes probabilités de présence de l'IA.



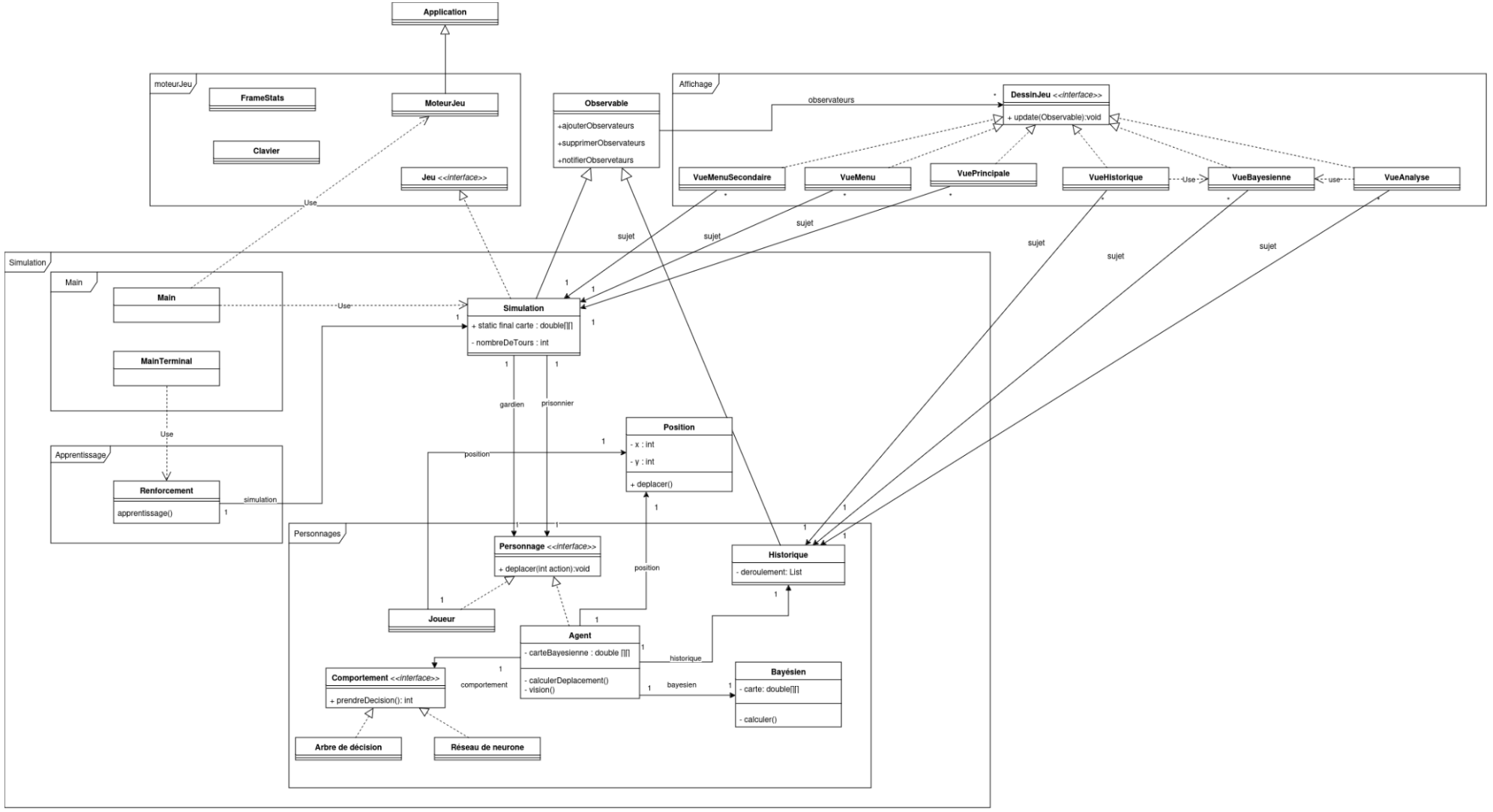
Vue de l'historique - Mode interactif

A chaque moment de chaque mode de simulation, l'utilisateur aura la possibilité d'afficher un menu pause qui stoppera la simulation. Il aura le choix de la reprendre ou de revenir au menu principal.



Menu de pause

- **Le diagramme de classe**



Package Simulation:

Le package “Simulation” se concentre sur la représentation des entités métiers de notre système et sur la partie apprentissage. La classe Simulation représente le domaine métier de notre application.

Package Affichage:

Dans ce package est présent toutes les vues que l’application finale possèdera

Package Moteur de Jeu:

Ce package concentre surtout la gestion des évènements clavier pour l’interaction de l’utilisateur avec le système. Et par la suite pour l’implémentation du temps réel, cette partie s’occupera de gérer la boucle principale d’affichage et le nombre de rafraîchissement par seconde.

Dans notre conception, nous avons utilisé le patron **Observateur** et nous nous sommes basés sur l’architecture **MVC**. Bien que les contrôleurs n’ont pas été représentés dans ce diagramme, nous allons les implémenter par la suite.

- **Le diagramme de séquence système**

Voici un diagramme de séquence système qui représente une interaction “classique” entre l’utilisateur et le système pour le déroulement d’une partie où l’utilisateur choisit d’incarner un agent.

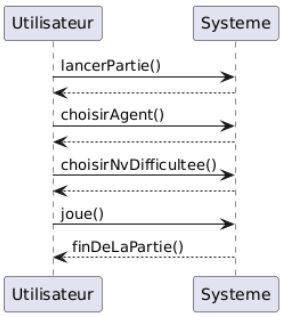


Diagramme de séquence n°1

Un autre diagramme de séquence système dans lequel le l'utilisateur choisit de ne pas incarner un rôle et seulement de laisser la simulation se dérouler en observant. Pour à la fin consulter le bilan de la simulation qui s'est déroulée.

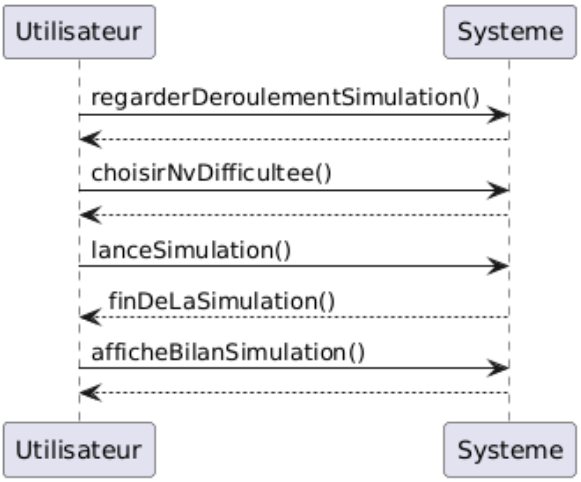


Diagramme de séquence n°2

Un diagramme dans lequel l'utilisateur choisit de quitter l'application en plein milieu de sa partie.

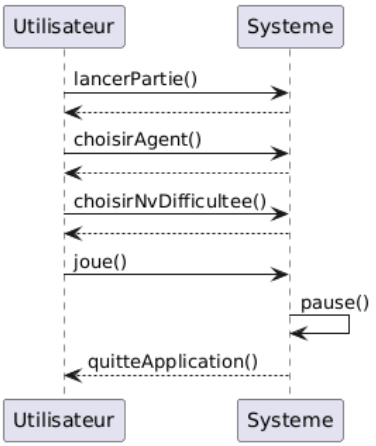


Diagramme de séquence n°3

- **Le diagramme de séquence du déplacement de l'agent**

  Voici le diagramme de séquence du déplacement d'un agent

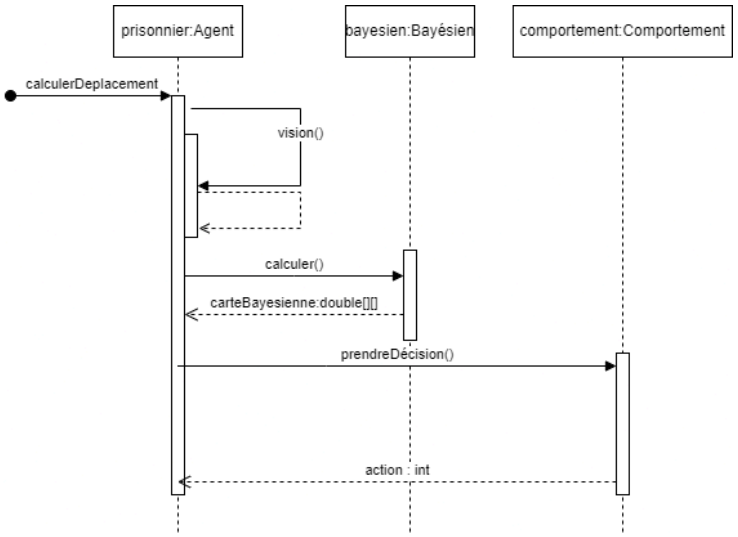


Diagramme de séquence du déplacement d'un agent

6. Gestion des risques

Dans cette partie, nous avons énuméré et classé tous les principaux risques de notre projet.

Risques	Explications	Statut
Qu'il n'y ait pas de comportement intelligent	Que contre un utilisateur, l'A entraînée soit de trop faible niveau.	Faible
Que l'une des IA mettent trop de temps de calcul	L'utilisateur devra attendre plus longtemps avant de pouvoir faire une action.	Faible
La simulation soit trop simple pour l'un des 2 agents	Pour le moment, ce risque est problématique car nous ne trouvons pas de solutions, mais ce risque est solvable.	Moyen
Que le temps d'apprentissage soit trop important	Que l'algorithme ne soit pas adapté, mal implémenté ou bien que les données soient trop complexes.	Fort

Grâce à ce tableau, nous avons pu voir les différents risques majeurs de notre projet pour pouvoir les classer par ordre d'importance. Nous serons donc beaucoup plus vigilants sur ces 4 risques.

7. Planning des itérations

Pour une meilleure efficacité, nous avons répartis au mieux les futures fonctionnalités que nous souhaitons concevoir.

➢ 1ère itération : (22h)

- Base du moteur de jeu → Célie,Luc
- Création du menu → Maëlle
- Création de la vue de la simulation interactive → Maëlle
- Mise en place de l'inférence bayésienne → Luc, Matias
- Création de l'arbre de décision du gardien → Célie,Matias
- Ajout des contrôles du prisonnier → Célie

➢ 2ème itération : (30h)

- Ajout du réseau de neurones sur le prisonnier → Luc, Matias
- Ajout du contrôle du gardien → Célie
- Implémentation d'un système d'analyse faire stat) → Maëlle
- Implémentation du lancement de l'application sans javaFx (vue terminal) → Matias
- Ajout des historiques (vues) → Célie

➢ 3ème itération : (30h)

- En fonction des résultats affinage des paramètres → Célie, Maëlle, Matias, Luc
- Évaluation de l'apprentissage pour en avoir un bon ajustement→ Célie, Maëlle, Matias, Luc

➢ <u>4ème itération :</u> (30h)

- Ajout du temps réel → Matias, Luc
- Ajout du menu de choix d'IA → Célie, Maëlle
- Ajouter les apparitions aléatoires dans la carte des personnages

➢ <u>5ème itération :</u> (20h)

- Temps pour les éventuels retards sur les fonctionnalités précédentes→ Célie, Maëlle, Matias, Luc
- Ajouter plusieurs cartes

➢ <u>6ème itération :</u> (20h)

- Optimisation et correction des erreurs → Célie, Maëlle, Matias, Luc
- Amélioration de l'affichage