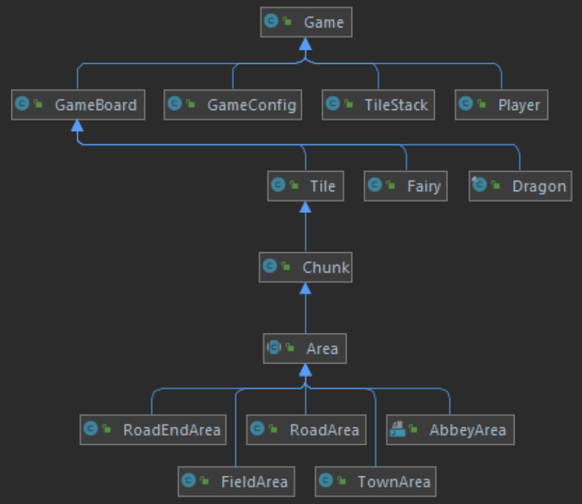
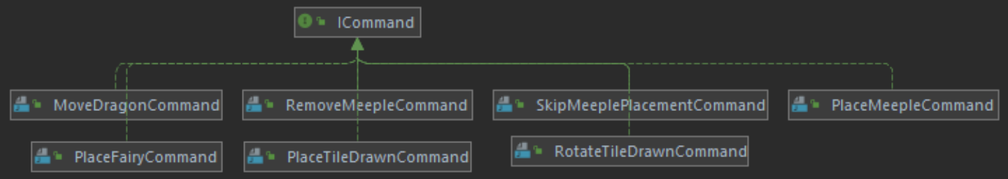
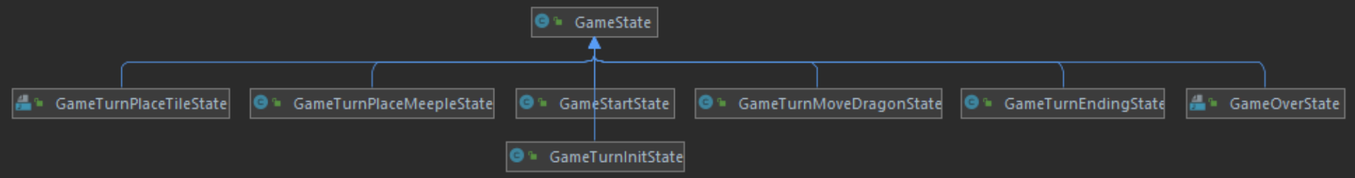
CARCASSONNE

UN JEU SERVEUR/CLIENT

AGLAE / BLANC / CHIAPPE /
LE CONTEL / RIHET







SOMMAIRE

I- FONCTIONNALITÉS

Bilan général des fonctionnalités fonctionnelles

II- CHOIX DE CONCEPTION

Justification des choix au travers des questions prédéfinies

III- ORGANISATIONS DES TESTS

Tout ne s'est pas passé comme prévu. Ce qui a freiné le développement

IV- GESTION DU PROJET

Ce que nous aurions pu faire

I- Fonctionnalités

LISTE DES PRINCIPALES FONCTIONNALITÉS

Support de toutes les règles du jeu de base

Support de l'extension "Princesse et Dragon"

Couleur configurable sur la console

Architecture Client / Serveur

Statistiques sur 500 parties

Name	Player ID 1	Player ID 2
POSITION	1	2
RESULTS (score)	131	95
ROAD POINTS	5	0
TOWN POINTS	102	89
ABBEY POINTS	9	0
FIELD POINTS	15	6
PARTISANS PLAYED	7	7
PARTISANS REMAINED	0	0

LES AXES

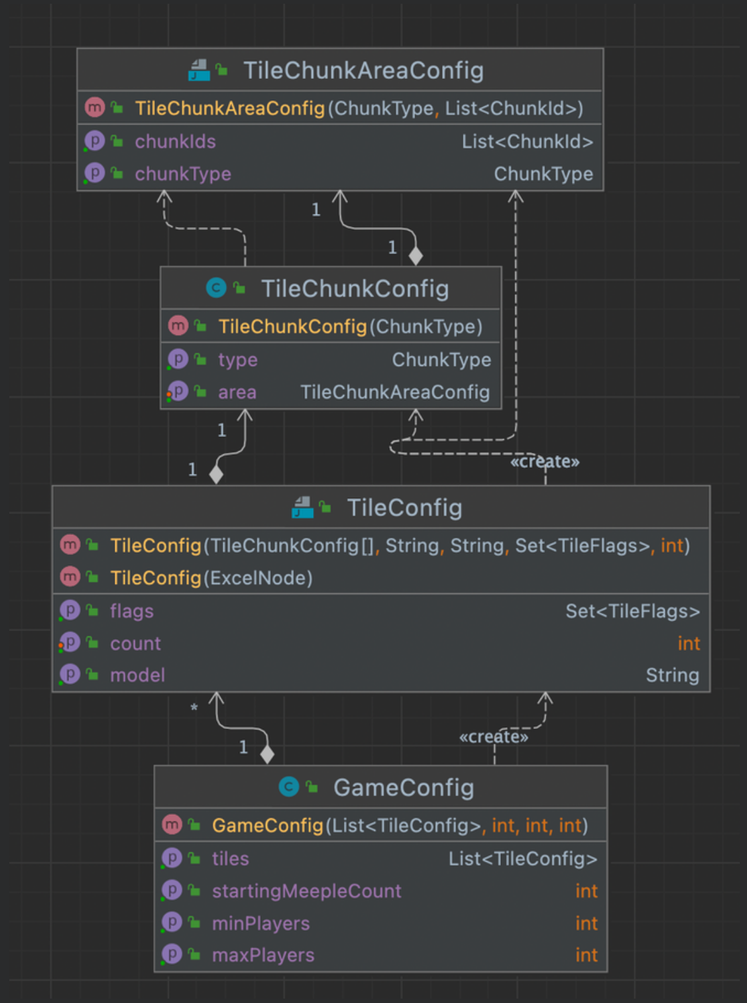
Configuration de l'environnement

Représentation du plateau

Architecture Client / Serveur

Conception de l'Intelligence Artificiel

CHOIX DE CONCEPTION



Configuration du jeu

Configuration de la logique

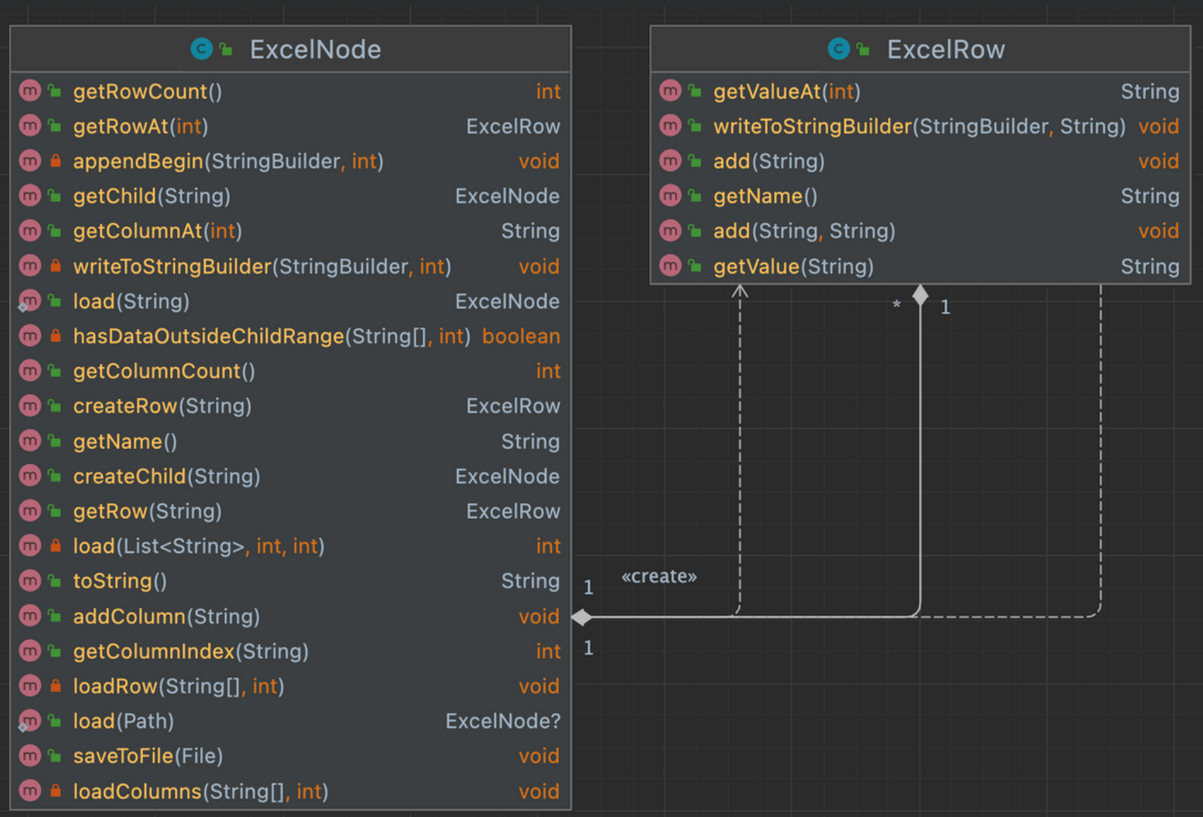
Configuration des joueurs

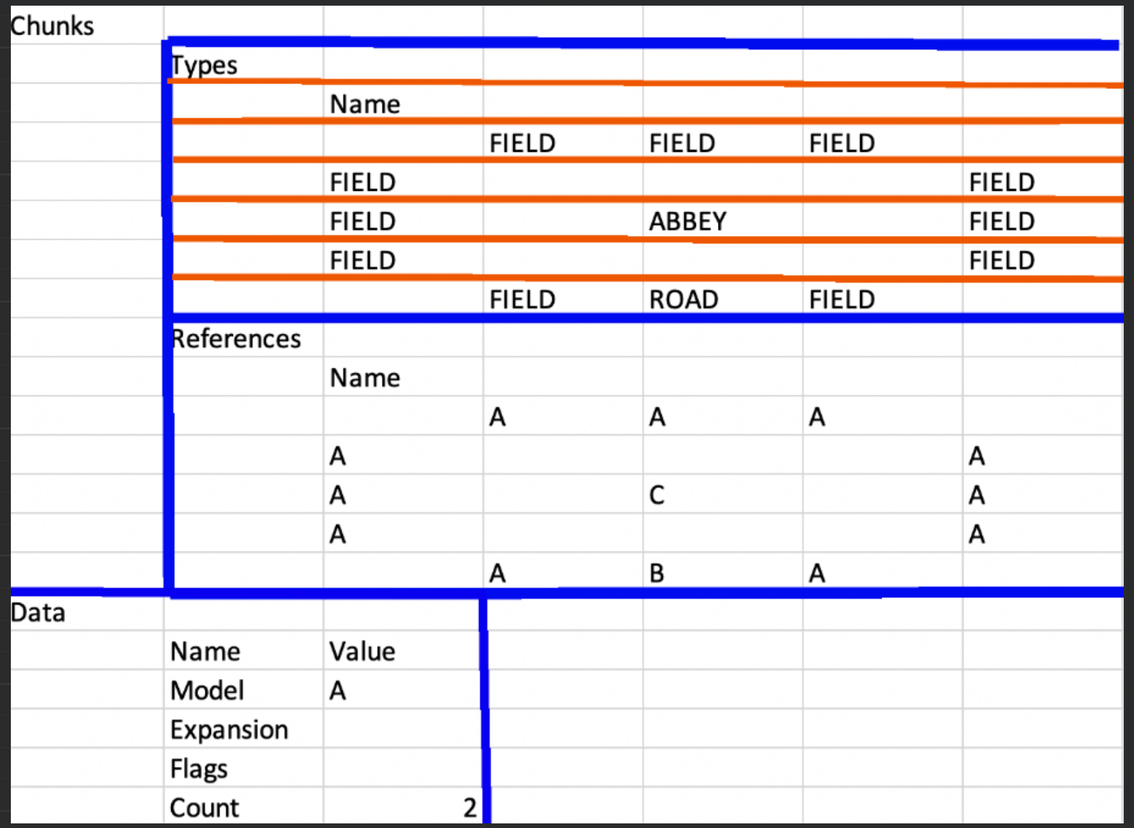
Name	Value
MinPlayers	2
MaxPlayers	5
StartingMeepleCount	7

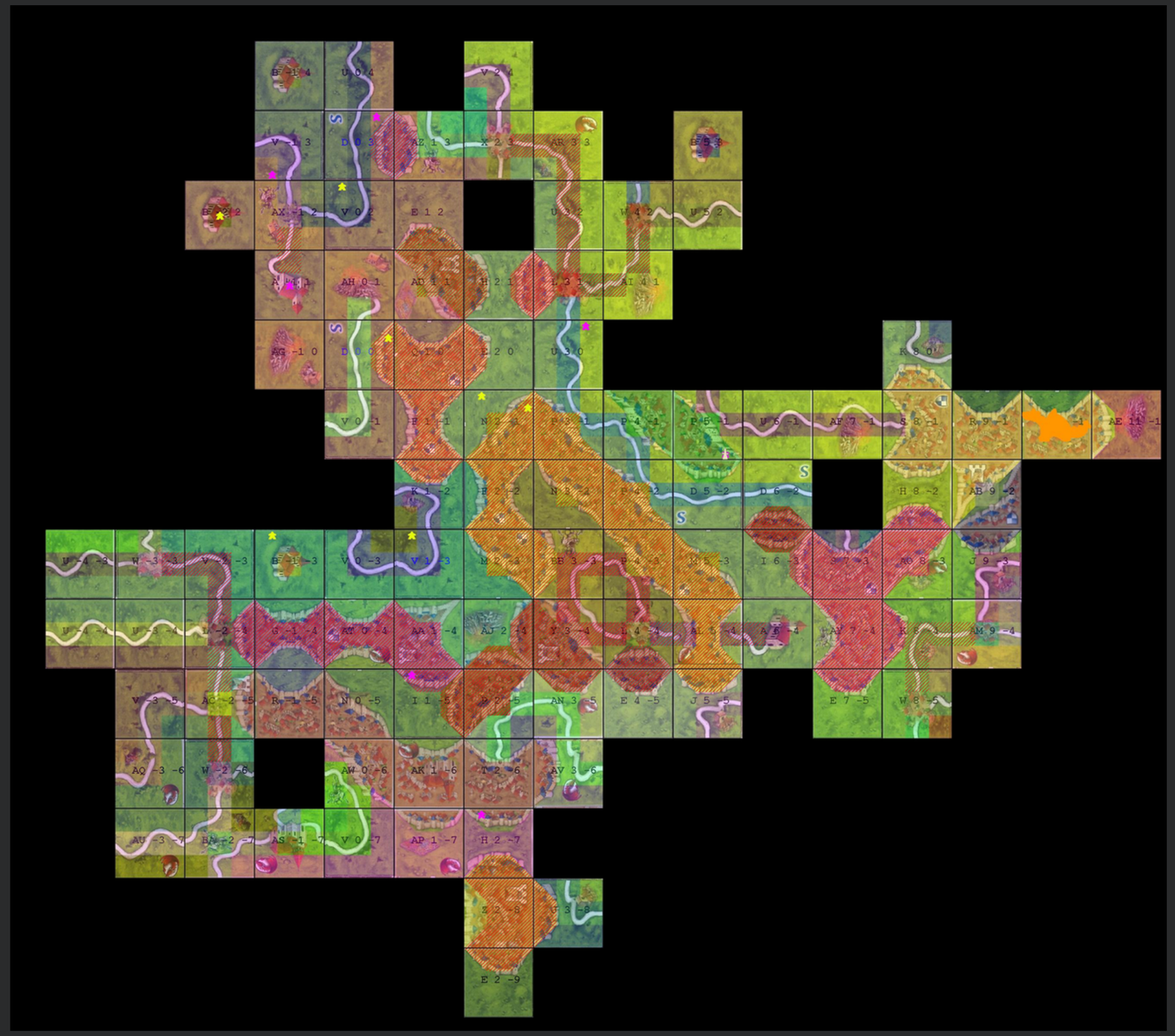
Configuration d'une tuile

Chunks						
	Types					
		Name				
			FIELD	FIELD	FIELD	
		FIELD				FIELD
		FIELD		ABBEY		FIELD
		FIELD				FIELD
			FIELD	ROAD	FIELD	
	References					
		Name				
			A	A	A	
		A				A
		A		C		A
		A				A
			A	B	A	
Data						
	Name	Value				
	Model	A				
	Expansion					
	Flags					
	Count	2				

Chargement des fichiers Excel

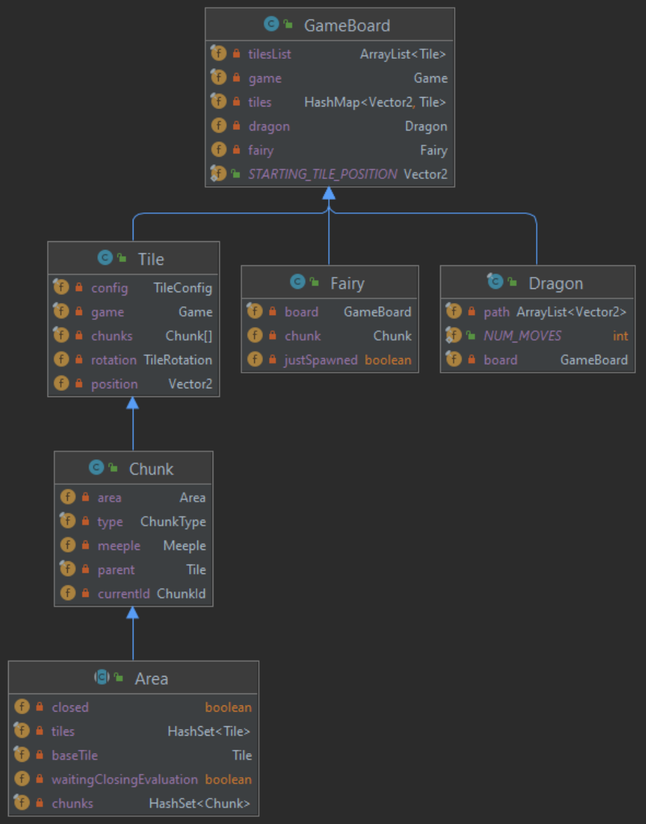




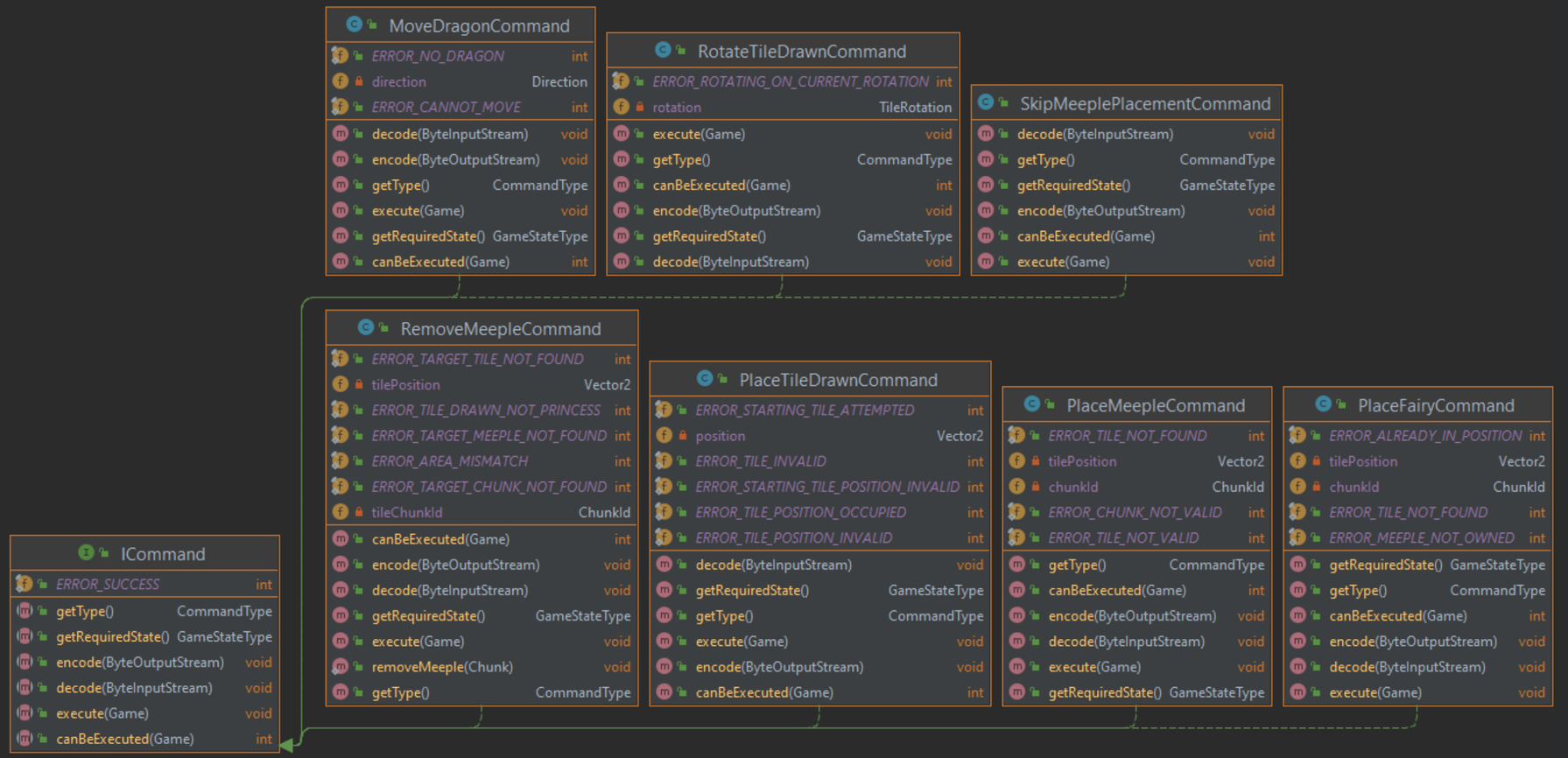


Logique de Jeu

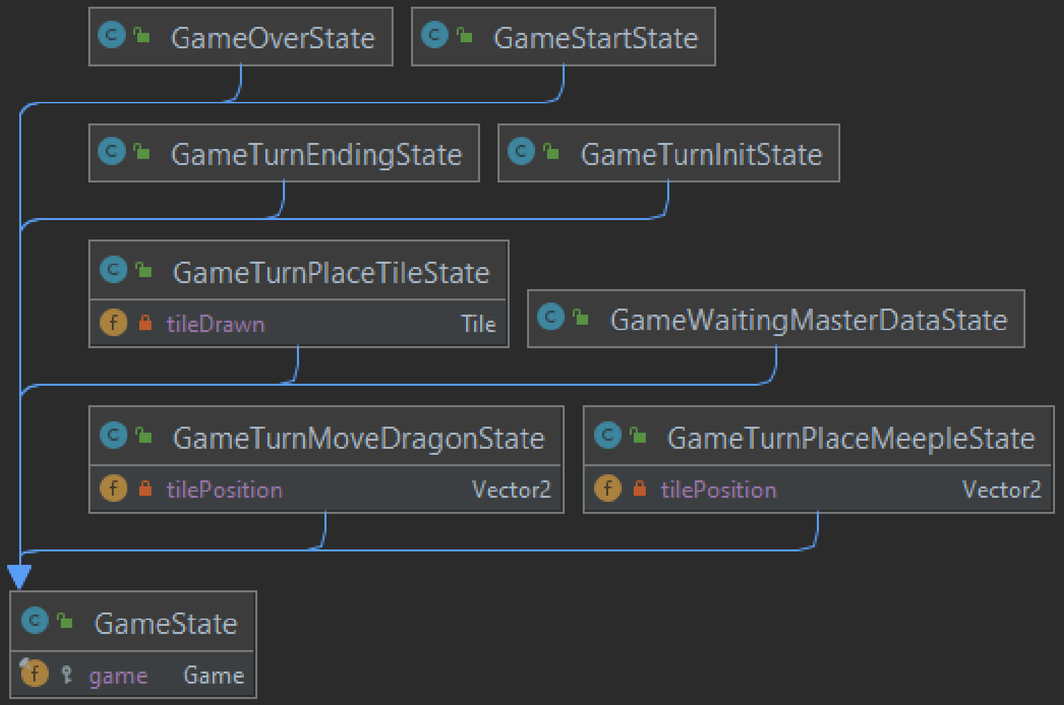
Logique de jeu
Plateau

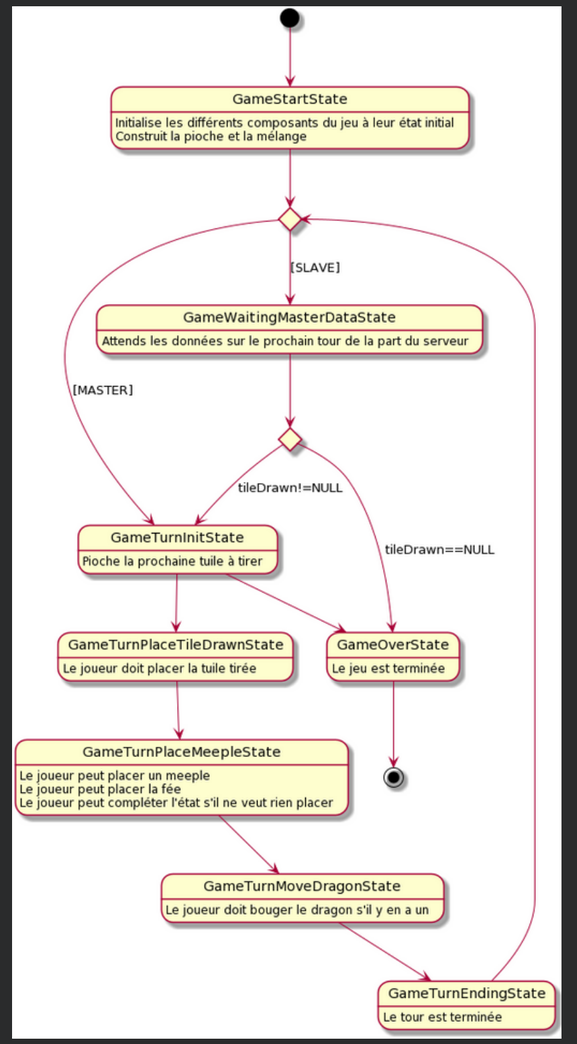


Logique de jeu
Commandes



Logique de jeu
Etats





CHOIX DE LA STRUCTURE

PROTOCOLE DE COMMUNICATION

ECHANGE ENTRE CLIENT/SERVEUR

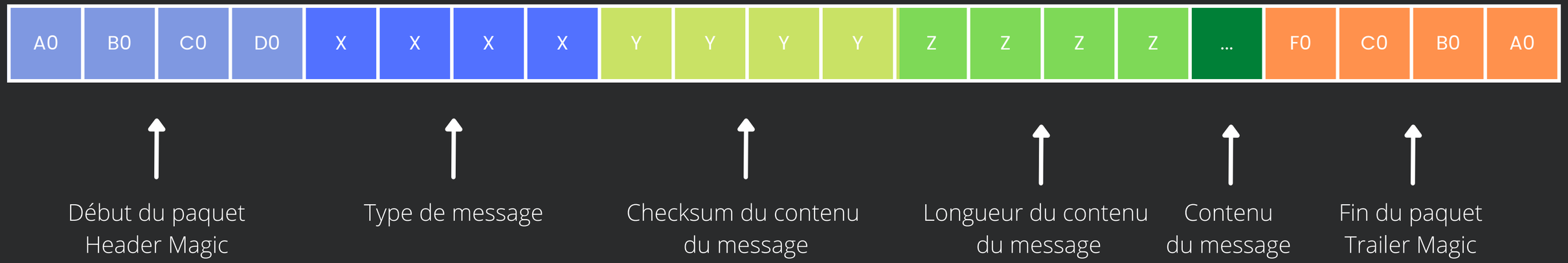
Client serveur

Choix de la structure

	Moteur de jeu sur le serveur	Moteur de jeu sur le client et le serveur
Avantage	- Client qui ne sait que ce qu'il a à savoir - Client indépendant de la logique de jeu	- Client qui sait tout de l'état actuel du jeu - Client indépendant du serveur pour simuler une action de l'IA afin de prévoir les conséquences d'une action - Client indépendant du serveur pour jouer au jeu (mode offline) - Economie de bande passante (pas besoin de renvoyer un instantané du plateau de jeu à chaque action du client)
Inconvénient	- Client qui n'a pas une vue complète sur l'état du jeu - Client fortement dépendant du serveur - Client qui a besoin du serveur pour simuler une action	- Client peut savoir des choses qui ne lui sont pas / peu utiles
Choix	Non	Oui

Protocole de jeu

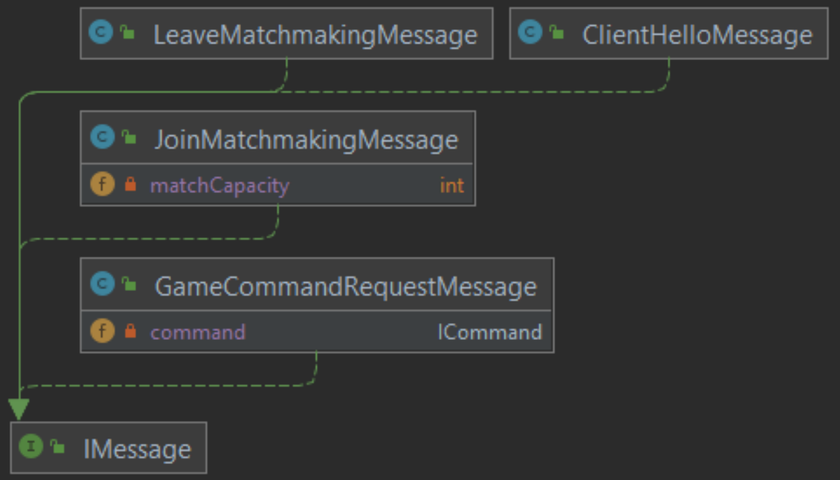
Paquet (protocole)



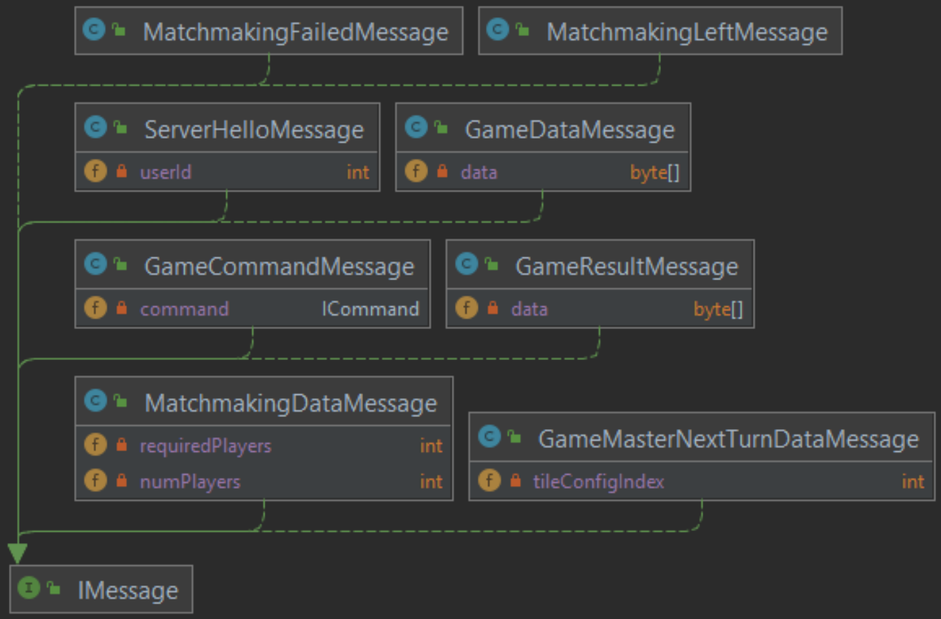
Protocole de jeu

Message

Messages du client

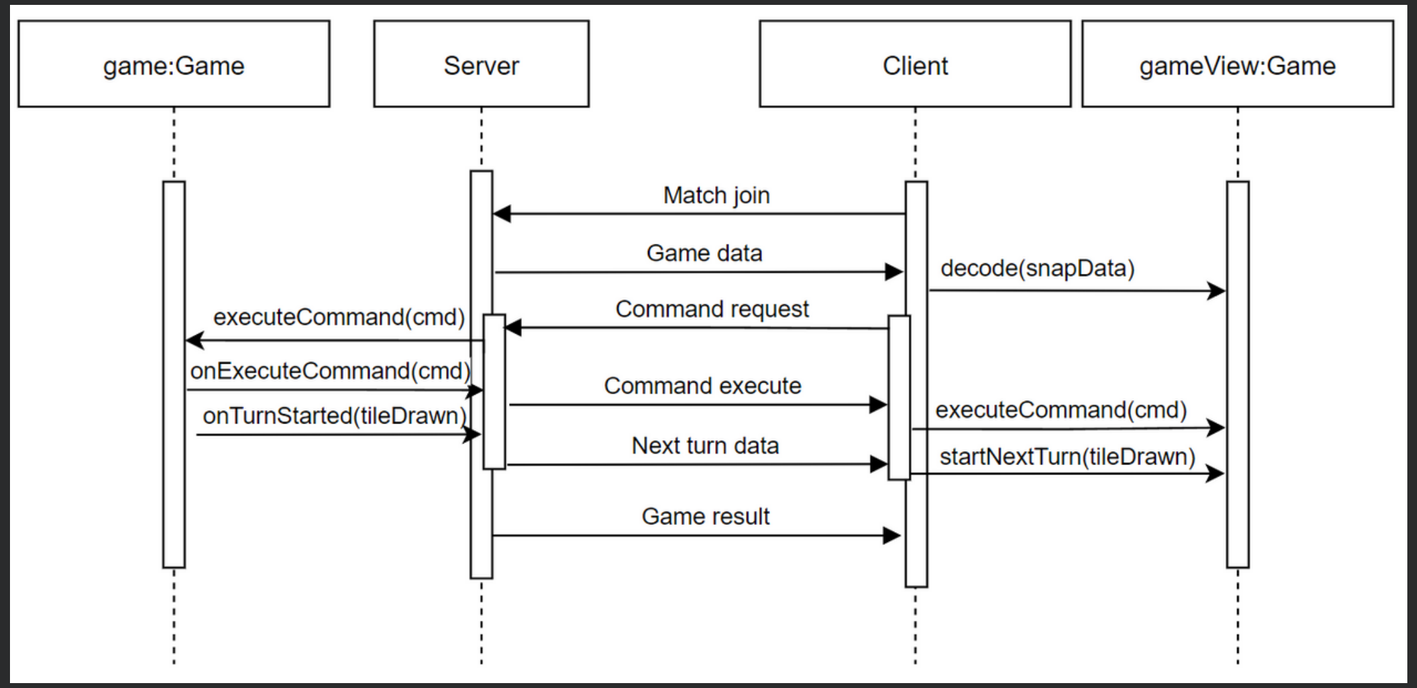


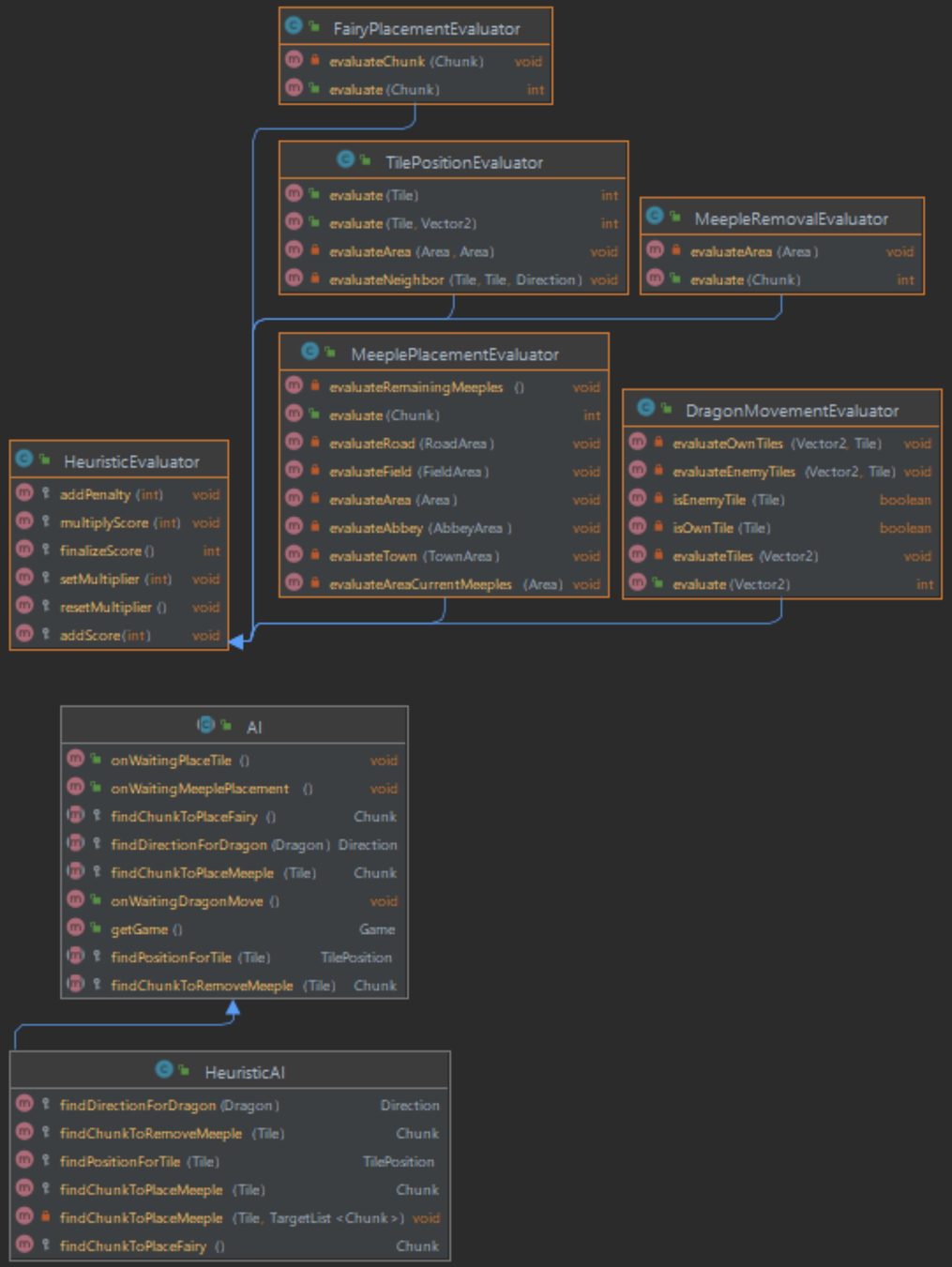
Messages du serveur



Protocole de jeu

Séquence d'échange simplifiée



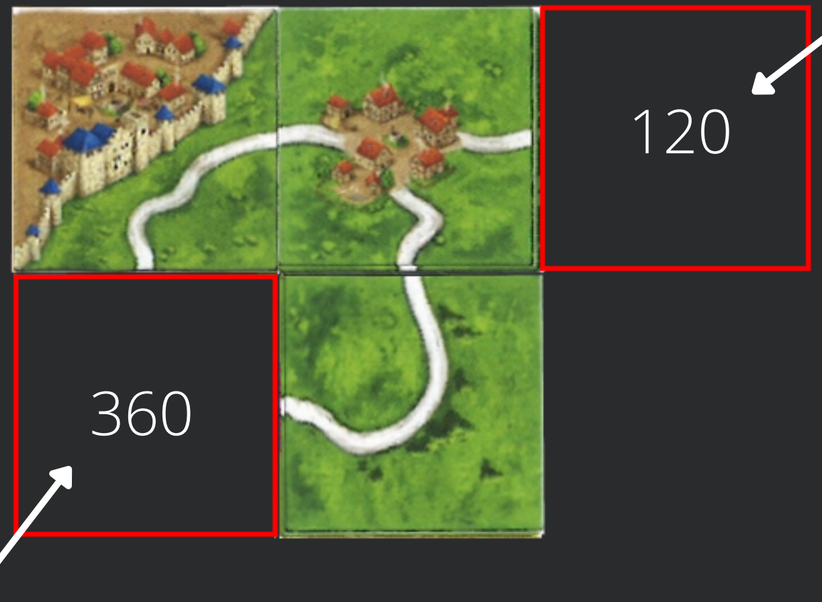


L'Intelligence Artificielle

L'Intelligence artificielle
Exemple de placement

Tuile à placer

Plateau de jeu



1 * [2 * (30 * 2) = 120

Multiplicateur Unique Zone | Multiplicateur Type Zone | Score zone presque fermée | Multiplicateur Un seul coté libre

2 * [2 * (30 * 3)] = 360

Multiplicateur Multi-zones | Multiplicateur Type Zone | Score zone presque fermée | Multiplicateur Zone fermée

```
assertFalse(gameBoard.isEmpty());
assertTrue(gameBoard.hasTileAt(tile.getPosition()));
assertEquals(tile, gameBoard.getTileAt(tile.getPosition()));
```

```
✔ Tests passed: 1 of 1 test – 22 ms

"C:\Program Files\Java\jdk-16.0.2\bin\java.exe" ...

Process finished with exit code 0
```

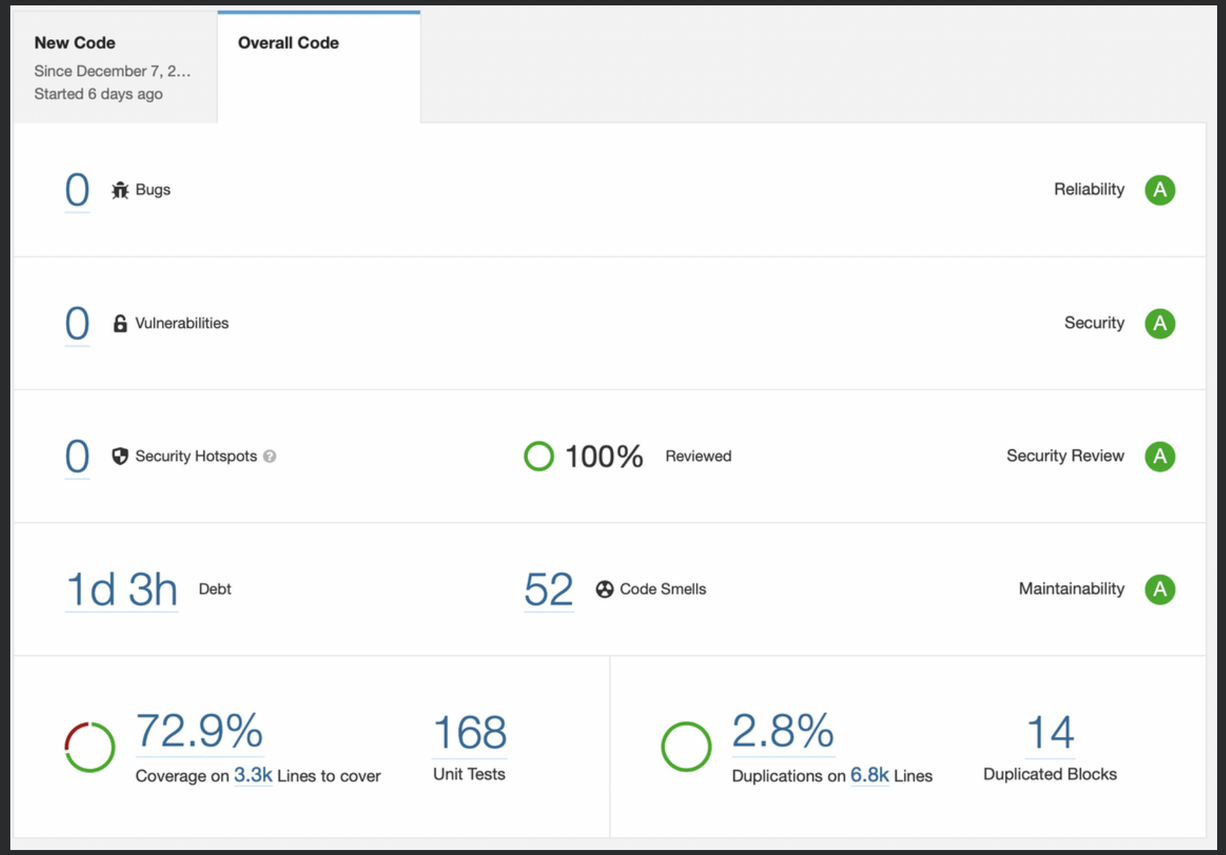
```java
@Test
void evaluateTown() {
    Tile tile1 = config.getTiles().stream().filter(t -> t.getModel().equals("E")).findFirst().get().createTile(game);
    Tile tile2 = config.getTiles().stream().filter(t -> t.getModel().equals("E")).findFirst().get().createTile(game);

    tile1.setPosition(new Vector2( x: 0,  y: 0));
    tile2.setPosition(new Vector2( x: 0,  y: 1));

    game.getBoard().place(tile1);

    TileRotation bestRotation = TileRotation.values()[0];
    int maxScore = Integer.MIN_VALUE;
    for (TileRotation tileRotation : TileRotation.values()) {
        tile2.setRotation(tileRotation);
        int score = tilePositionEvaluator.evaluate(tile2);
        if (maxScore < score) {
            maxScore = score;
            bestRotation = tileRotation;
        }
    }

    assertEquals(TileRotation.DOWN, bestRotation);
}
```

Organisation des tests

168 tests

COMMENT ON A FAIT NOS TESTS

Gestion du projet



Gestion de projet

Lacune sur la couverture des tests

	Coverage	Uncovered Lines	Uncovered Conditions
carcassonne-client/src/main/java/client/ai/HeuristicAI.java	11.3%	52	42
carcassonne-client/src/main/java/client/service/GameStatisticsService.java	15.5%	36	13
carcassonne-client/src/main/java/client/service/MatchmakingService.java	21.1%	22	8
carcassonne-server/src/main/java/server/session/ClientSession.java	33.3%	18	8
carcassonne-server/src/main/java/server/matchmaking/Matchmaking.java	61.0%	11	5
carcassonne-client/src/main/java/client/ai/evaluator/MeeplePlacementEvaluator.java	62.3%	9	11
carcassonne-server/src/main/java/server/network/ClientConnection.java	65.6%	23	9
carcassonne-server/src/main/java/server/matchmaking/Match.java	65.9%	17	13
carcassonne-client/src/main/java/client/network/ServerConnection.java	66.7%	20	4
carcassonne-common/src/main/java/logic/command/PlaceFairyCommand.java	67.5%	8	5
carcassonne-common/src/main/java/logic/state/turn/GameTurnPlaceMeepleState.java	68.4%	5	1

Gestion de projet

Une fonctionnalité manquante

Abbaye dans la ville : Si vous décidez de poser un meeple sur cette tuile, vous devez choisir soit de le poser dans la ville, dans l'abbaye, ou dans le pré. Si vous le désirez, vous pouvez coucher le meeple dans l'abbaye pour distinguer ce moine des chevaliers qui occupent la ville. L'abbaye est complétée lorsqu'elle est entourée de tuiles, même si la ville est encore incomplète. Vous pouvez placer un moine dans cette abbaye alors que des chevaliers sont déjà présents ailleurs dans la même ville. De même, un moine posé dans cette abbaye n'empêche pas les joueurs de poser un chevalier dans la même ville.