

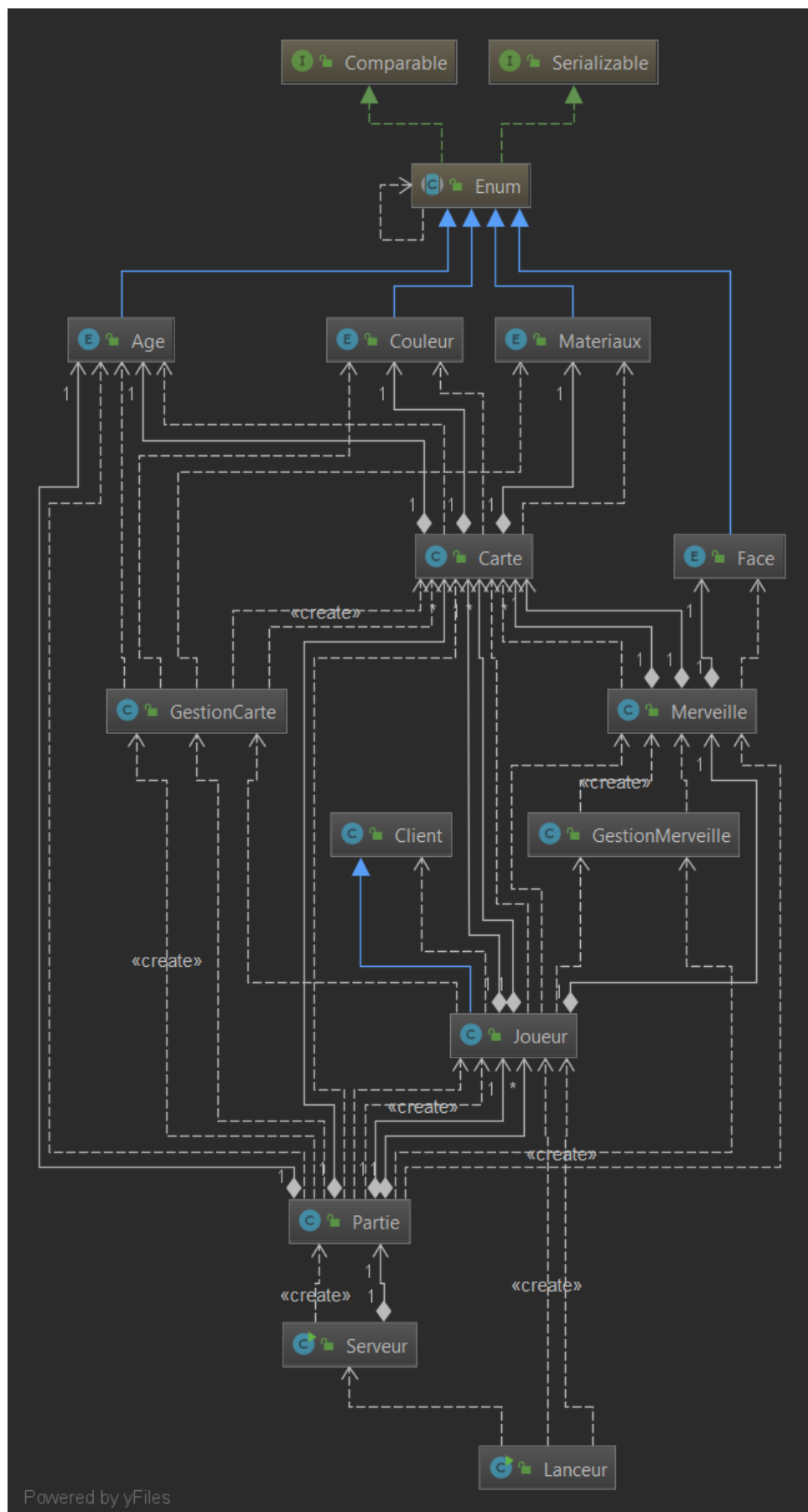
DIAGRAMME DE CLASSE

Le diagramme de classe qui va suivre représente les classes et les interfaces du jeu que nous développons : Seven Wonders (voir figure 1).

Les éléments du diagramme sont :

- Une classe Carte : qui va attribuer aux cartes plusieurs caractéristiques dont la couleur, la face, le matériau, l'effet (il peut y en avoir plusieurs), l'âge correspondant.
- Quatre énumérations qui vont correspondre aux caractéristiques précédentes, c'est-à-dire la couleur, la face, le matériau, l'effet et l'âge.
- Une classe Merveille : elle va être caractérisée par un nom, un identifiant et une face attribuée.
- Une classe Partie : liste des joueurs, distribution des merveilles, gestion des deck, golds, et le score.
- Une classe Joueur : qui prend en compte le nom et l'url pour identifier un joueur après la connexion au serveur, ensuite elle déclare la réception des merveilles et des cartes aux joueurs.
- Une classe Serveur : qui va servir de relation aux différentes classes, et qui va permettre aux joueurs de s'identifier, permettre de lancer une partie et ainsi de suite avec la distribution des cartes.
- Une classe Lanceur : qui va permettre de générer une partie dès que le serveur l'aura signalé (quand les joueurs seront connectés).

Figure 1 : Diagramme de classe en résumé



1) Éléments du jeu

La classe Carte :

Elle est constituée de plusieurs caractéristiques, les suivantes :

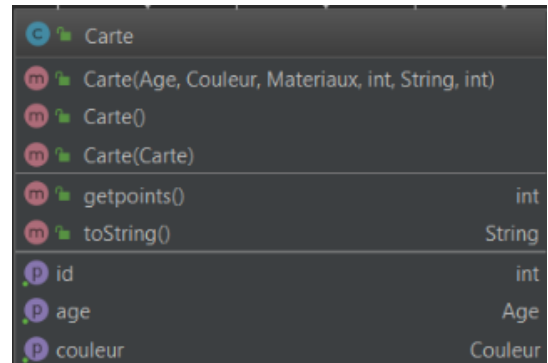
- Age (à partir de l'énumération Age)
- Couleur (à partir de l'énumération Couleur)
- Nom (String)
- Cout (Matériaux en Integer)
- Point de victoire (Integer)

Elle est constituée de plusieurs méthodes :

- Getpoints() de type Integer
- toString() de type String

Elle est constituée de 3 constructeurs :

- Carte(Age, Couleur, Matériaux, int, String, int)
- Carte()
- Carte(Carte)



c	Carte	
m	Carte(Age, Couleur, Matériaux, int, String, int)	
m	Carte()	
m	Carte(Carte)	
m	getpoints()	int
m	toString()	String
p	id	int
p	age	Age
p	couleur	Couleur

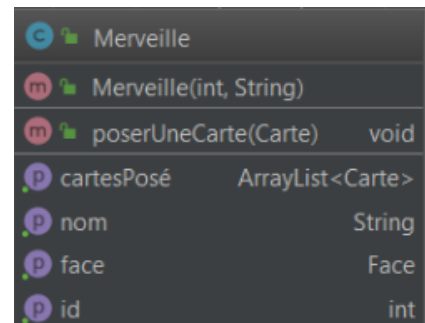
La classe Merveille :

Elle est constituée de plusieurs caractéristiques, les suivantes :

- cartesPosé (Array List<Carte>)
- ID (Integer)
- Face (à partir de l'énumération Face)
- Nom (String)

Elle est composée de deux méthodes :

- Merveille(int, String) : qui va permettre de créer un objet Merveille caractérisé par son nom et le nombre de construction.
- poserUneCarte(Carte) : cette méthode va permettre de prendre en paramètre une carte instanciée, et de l'ajouter à la liste des cartes posées (cartesPosés Array List).



c	Merveille	
m	Merveille(int, String)	
m	poserUneCarte(Carte)	void
p	cartesPosé	ArrayList<Carte>
p	nom	String
p	face	Face
p	id	int

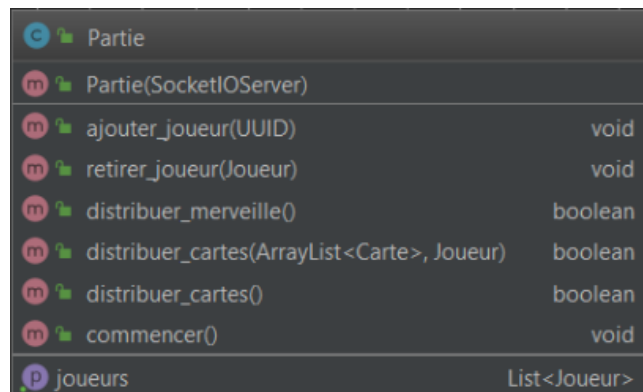
2) Création d'une partie

La classe Partie :

Elle est composée d'une liste de joueurs.

Elle est composée de plusieurs méthodes :

- Partie(SocketIOServeur) : qui permettra la création d'une partie pour en prenant en paramètre un serveur.
- Ajouter_joueur(UUID) : elle permet d'instancier un nouveau joueur.
- Retirer_joueur(Joueur) : elle permet de supprimer un joueur de la liste.
- Distribuer_merveille() : elle permet d'attribuer de façon aléatoire une merveille à chacun des joueurs.
- Distribuer_cartes(ArrayList<Carte>, Joueur) : elle permet d'attribuer de façon aléatoire un deck à un joueur choisi, autrement dit, un ensemble de cartes.
- Commencer() : cette méthode va faire appel aux méthodes citées ci-dessus pour permettre le lancement de la partie.



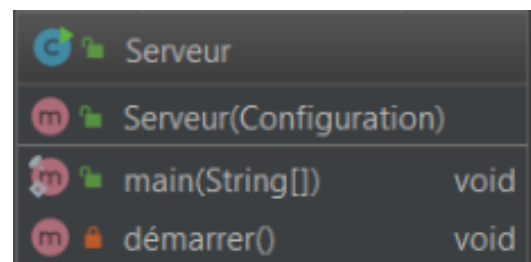
Partie	
Partie(SocketIOServeur)	
ajouter_joueur(UUID)	void
retirer_joueur(Joueur)	void
distribuer_merveille()	boolean
distribuer_cartes(ArrayList<Carte>, Joueur)	boolean
distribuer_cartes()	boolean
commencer()	void
joueurs	List<Joueur>

3) Lancement d'une partie

La classe Serveur :

Elle possède plusieurs méthodes, les suivantes :

- Serveur(Configuration) : qui va permettre d'instancier une nouvelle partie, et le serveur SocketIO.
- Main(String[]) : qui va permettre d'afficher les informations utiles quant à l'avancement de la partie.
- Démarrer() : qui va permettre la connexion du serveur, et ainsi de débiter la partie.

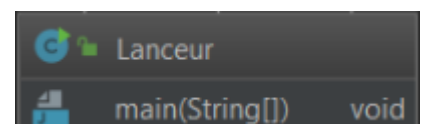


Serveur	
Serveur(Configuration)	
main(String[])	void
démarrer()	void

La classe Lanceur :

Cette classe possède une seule méthode main(String[]).

Cette méthode va servir à créer des threads, autrement dit les joueurs. Ensuite elle va établir la connexion entre les classes Serveur et Joueur, pour connecter les joueurs à la partie. Dès lors, le serveur sera lancé et prêt à lancer le jeu.



Lanceur	
main(String[])	void

Diagramme de classe en détail

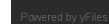
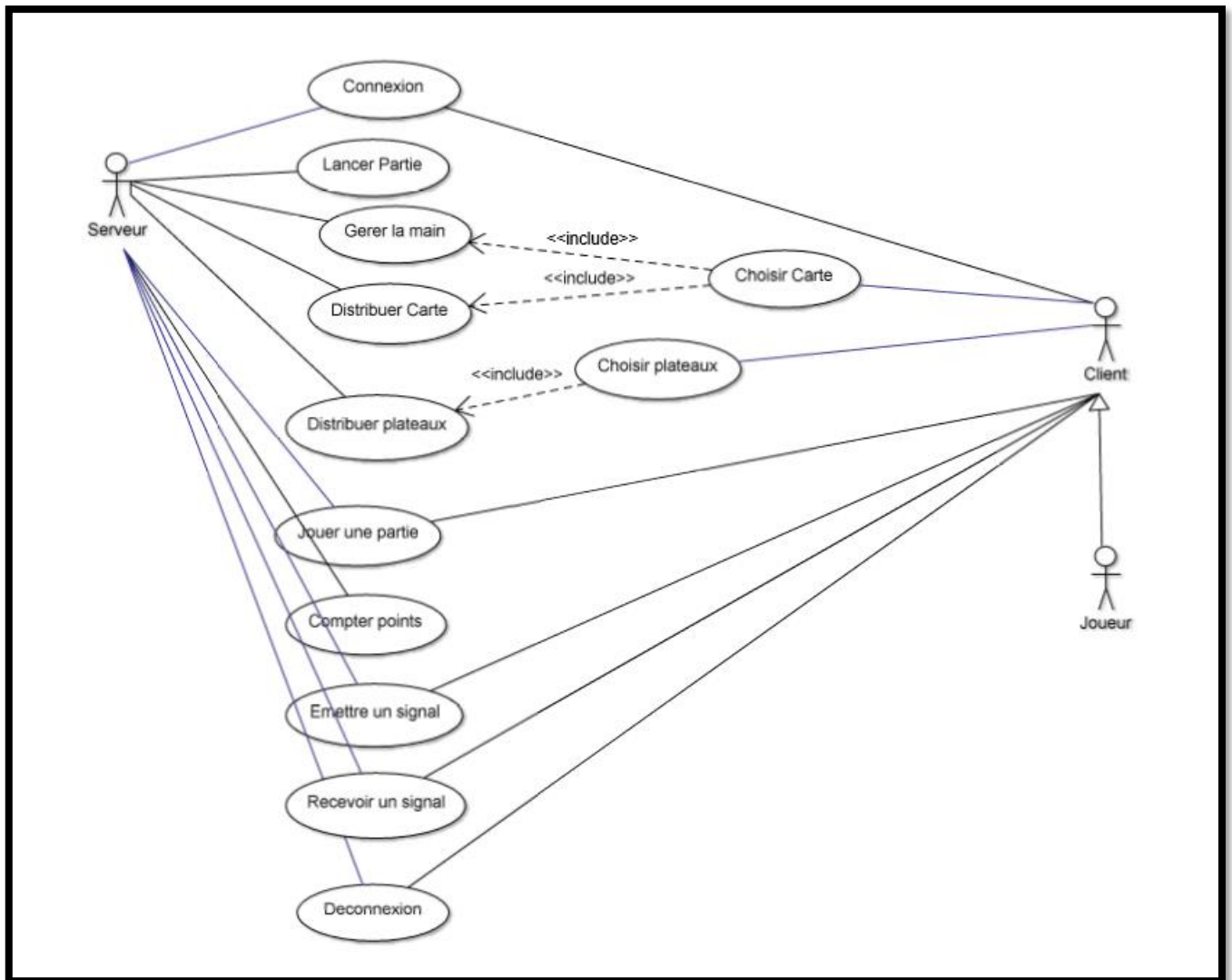


DIAGRAMME USE CASE



Serveur

Le Serveur est capable de permettre au client de se connecter grâce aux signaux qu'il aura reçu du client de la connexion.

Il peut donc lancer la partie puis distribuer les cartes (merveilles et ressources) et aussi les plateaux.

Il pourra permettre les joueurs de jouer. Une fois la partie terminée, il attend que le joueur lui lance un signal pour compter les points, après finir la partie et permettre la déconnexion des joueurs.

Client

Les clients vont d'abord se connecter au serveur ensuite émettre un message, en indiquant qu'ils sont bien connectés ainsi que leurs noms.

Ils vont recevoir un signal du serveur qui va leur annoncer qu'il a attribué une merveille et les autres cartes.

Les clients vont ensuite attendre un autre signal du serveur leur permettant de commencer à poser leurs cartes.

Une fois la carte posée ils transmettront un message de fin de tour et le serveur distribuera les cartes suivantes (selon le sens du jeu) de l'un de leurs voisins de la part du serveur.

Une fois que le client n'aura plus qu'une carte en sa possession, le serveur la récupèrera et commencera l'âge suivant.

Après le troisième âge le serveur calculera leur score et déclarera un vainqueur.

A la fin il permettra aux clients de se déconnecter du serveur.

DIAGRAMME D'ACTIVITE

Diagramme d'Activité du déroulement d'un âge avec une ressource

- ➔ Tout d'abord, les clients commencent par se connecter en donnant les noms qu'ils vont utiliser en tant que joueur.
- ➔ Le serveur les enregistre en les ajoutant par la suite dans la partie si et seulement s'il y a plus de 3 joueurs (sinon la partie n'est pas lancée).
- ➔ Une fois les joueurs ajoutés, le serveur commence la partie en distribuant les cartes (C'est à dire les cartes que les joueurs vont utiliser), les cartes merveilles et les cartes ressources. Avant le positionnement des joueurs (gauche et à leur droite), le serveur lance la partie
- ➔ Après le lancement de la partie, les joueurs choisissent la carte qu'ils désirent poser, ou défausser en fonction de leurs ressources.
- ➔ Le serveur va attendre de recevoir un signal lui communiquant la fin des actions des joueurs pour pouvoir choisir s'il enclenche un nouveau tour (en faisant tourner les decks de cartes) ou si l'âge est terminé (s'il y a eu plus de 3 tours).
- ➔ A la fin de la partie, il affiche les scores et termine ses actions.

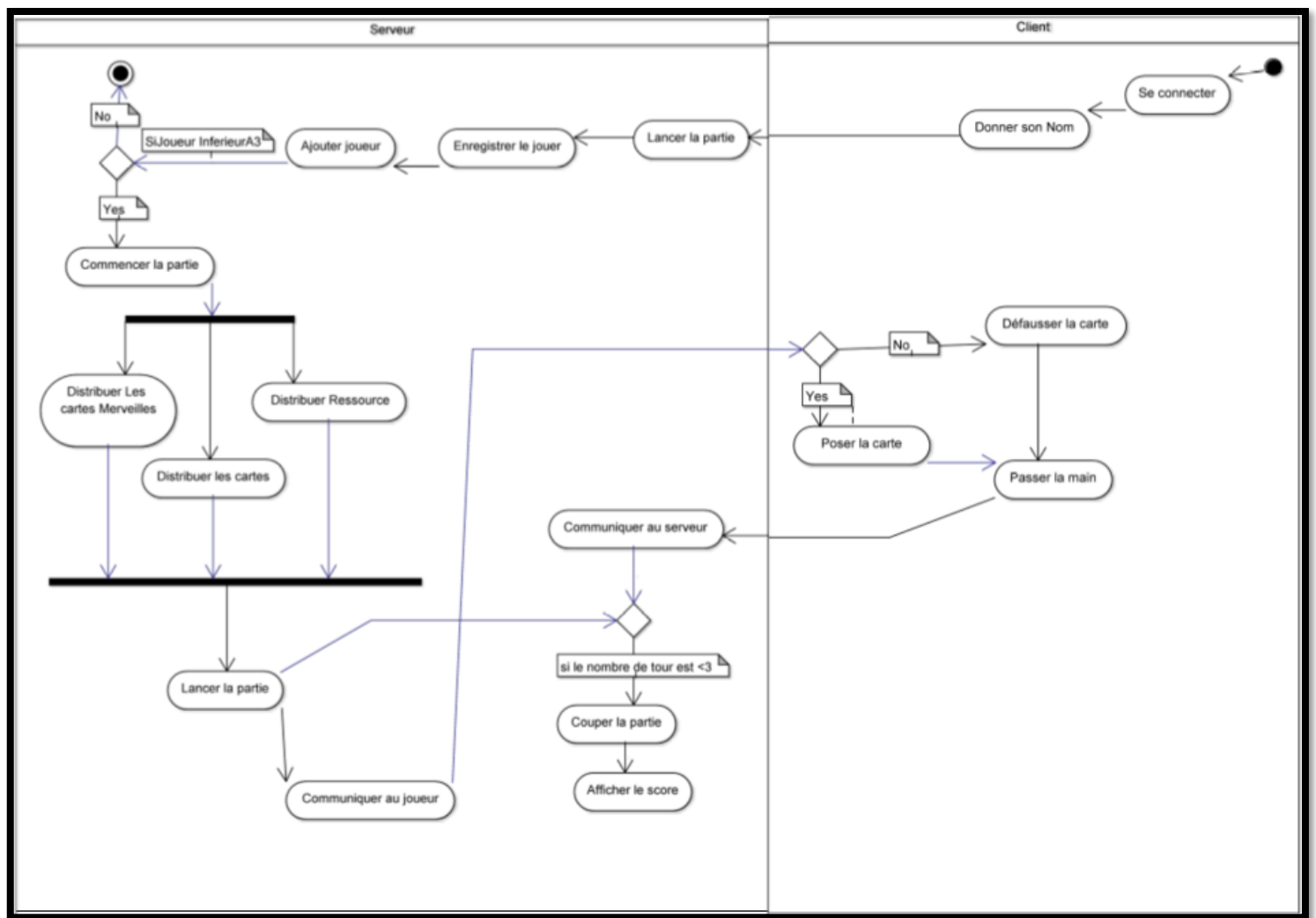


DIAGRAMME DE SEQUENCE

Diagramme séquence Client-Serveur

Le diagramme proposé décrit les étapes lors de la communication entre le client et le serveur, les étapes sont les suivantes :

- Le client se connecte tout d'abord au serveur avec la méthode `seConnecter()`.
- Le serveur pour donner suite à la connexion d'un client va créer les écouteurs d'évènements puis établir la connexion avec le client
- À la suite de cela, différents messages peuvent être envoyés entre le client et le serveur
- A la fin de l'échange des messages, le client se déconnecte puis envoie un message dans lequel est inscrit son attente de déconnexion
- Le serveur déconnecte alors le client

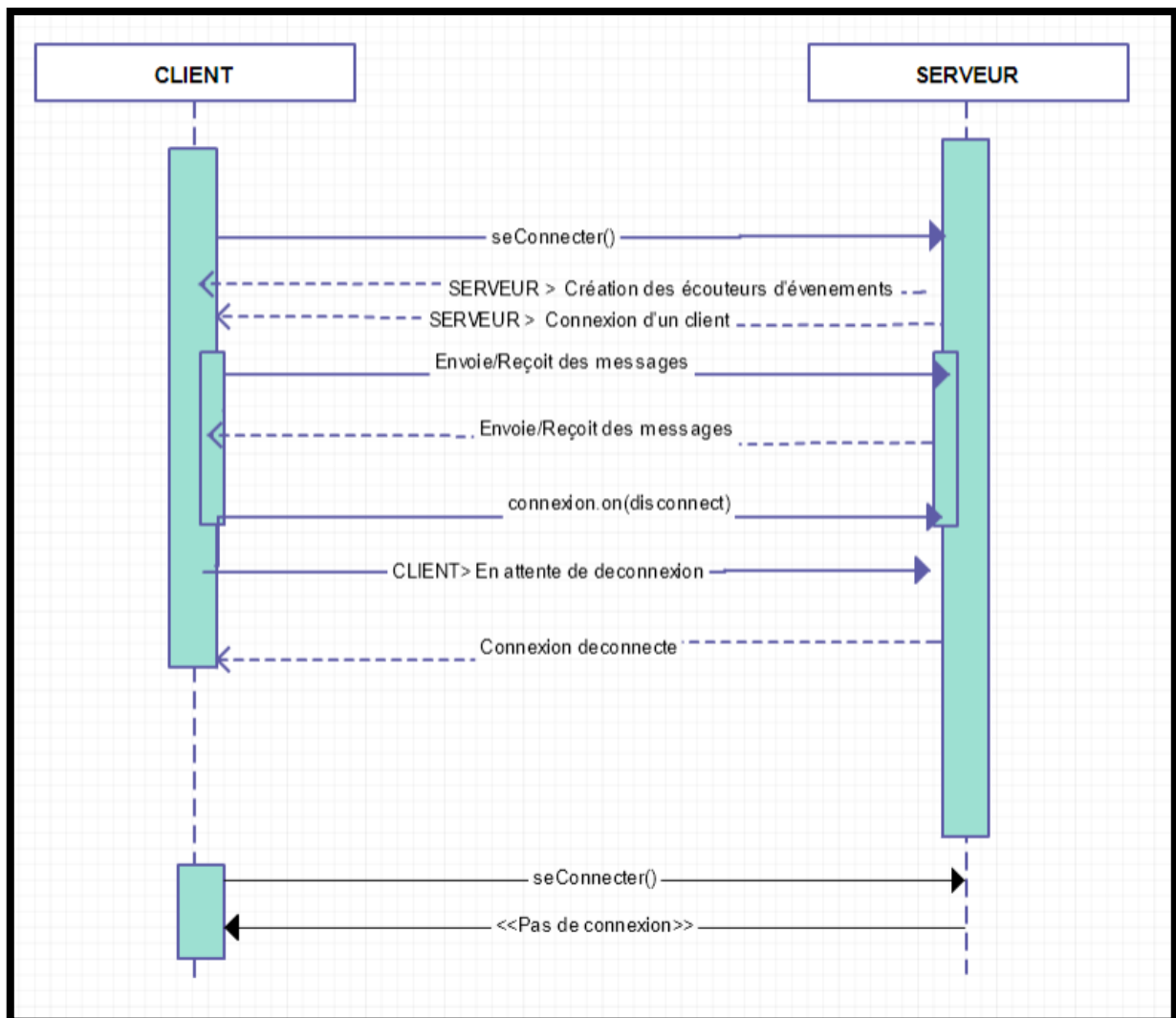


Diagramme séquence Partie-Serveur

Le diagramme proposé décrit les étapes lors de la communication entre la partie et le serveur, les étapes sont les suivantes :

- Lorsque la partie est créé, on ajoute un évènement poser une carte, le serveur retourne alors la carte posée par les tous joueurs avec le biais d'une boucle
- Ensuite vient la configuration de la partie, on commence tout d'abord par ajouter les joueurs.
- Puis, on lance la partie.
- Viens ensuite la distribution des cartes et des merveilles qui seront réceptionnées par tous les joueurs.

