7wonders

Rapport COO #2

Nos diagrammes ont été réalisés avec :

Draw.io

StarUML

Visual Paradigm

EL HAROUI Jassim – DRISSET Nicolas – GASMI Zakaria

Encadré par : Mme. CABRIO et M. RENEVIER

Table des matières

I. Introduction

Au cours de notre année en L3 Miage à l'Université de Nice Sophia Antipolis, nous devions réaliser un projet JAVA Client/Serveur pour le Jeu de 7wonders qui a été proposé par M. RENEVIER.

Suite à des difficultés rencontrées qui ont fortement impacté le premier rendu du rapport COO, ce document a pour but de bien expliquer l'architecture et le fonctionnement de notre projet dans un premier temps grâce à un glossaire et à un diagramme d'activité du projet qui représente le fonctionnement global de l'application et qui représente l'interaction globale entre le serveur et les clients. Puis des diagrammes Use Case qui montrent les cas d'utilisation et permet de cerner les différentes fonctionnalités, ainsi que des diagrammes de séquences qui permettent de bien montrer les messages échangés entre les lignes de vie, présentés dans un ordre chronologique, et finalement les diagrammes de classes qui détaillent les packages, les classes, leurs relation et fonctionnement.

II. Glossaire

Socket côté client : permet de se connecter à une machine distante afin de communiquer avec elle.

Socket côté serveur : connexion qui attend qu'un client vienne se connecter afin de communiquer avec lui.

Diagramme de cas d'utilisation : permet d'identifier les possibilités d'interaction entre le système et les acteurs.

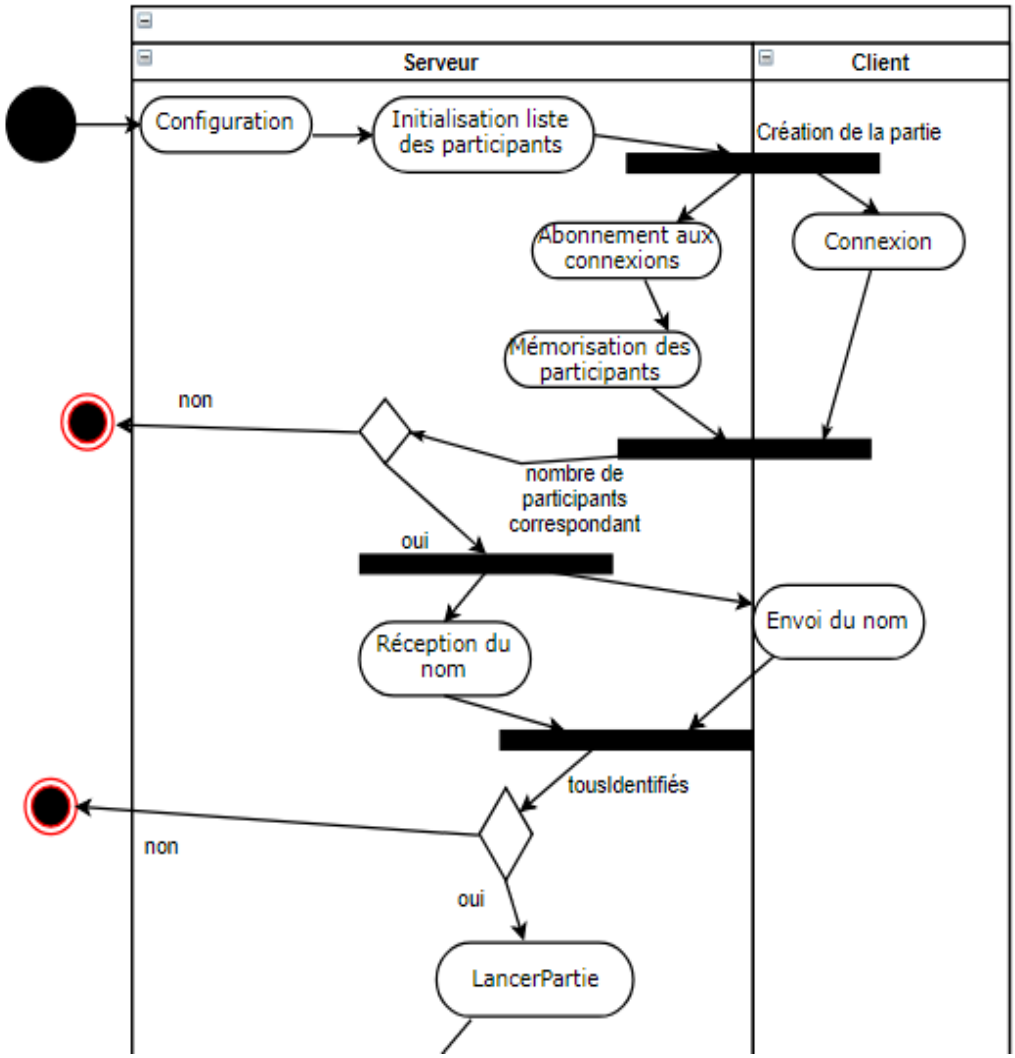
Diagramme d'activité : permet de modéliser le comportement du système, dont la séquence des actions et leurs conditions d'exécution.

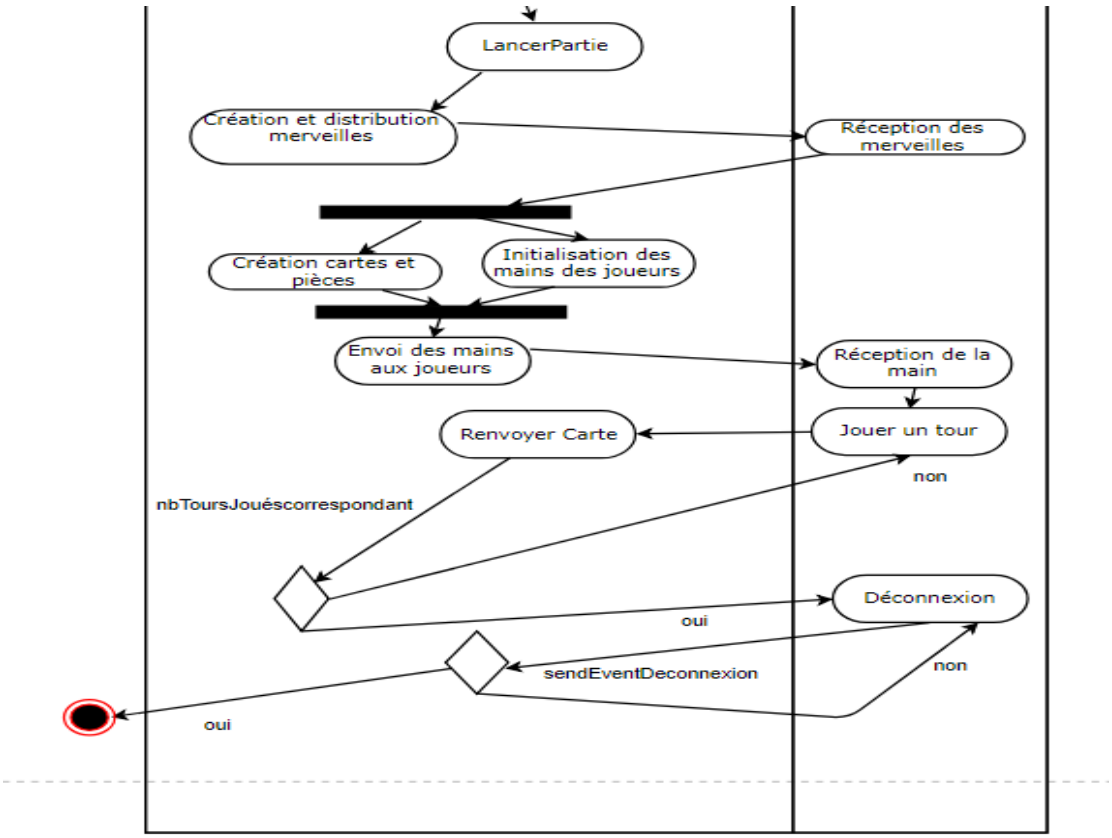
Diagramme de séquence : permet de bien montrer les messages échangés entre les lignes de vie, présentés dans un ordre chronologique

Diagramme de classe : permet de décrire clairement la structure en modélisant les classes, leurs attributs, leurs opérations et les relations entre eux.

Booléens : variables importantes aussi bien au niveau de la conception qu'au niveau de la programmation, prends les valeurs vrai ou faux, et permet de définir un prédicat servant à la suite de l'exécution d'un programme.

III. Diagramme d'activité





Ce diagramme représente l'interaction entre le serveur et les clients. Dans un premier temps, le serveur lance sa configuration puis crée la partie tout en initialisant une liste de joueurs. La création de la partie est lancé et le serveur s'abonne aux connexions des clients qui se déroulent en parallèle et les ajoute au fur et à mesure dans la liste des joueurs, d'où la mémorisation.

Le nombre de connexions représente le nombre de participants et doit correspondre en étant inférieur ou égal au nombre de joueurs défini dans la configuration du jeu, sinon il y a sortie. Les joueurs enregistrés envoient aussi leur nom et le serveur les identifie en parallèle. Si tous les joueurs enregistrés ont été identifiés, en se servant du booleen **tousIdentifies** , le serveur lance la partie nouvellement créée et initialise les merveilles pour les distribuer aux joueurs . Les clients reçoivent la merveille puis le serveur initialise ensuite les cartes et les associe en parallèle à une main de joueur, puis le serveur envoie les différentes mains aux joueurs.

Il parcourt la liste des joueurs pour jouer un tour. Tant que le nombre de tours est inférieur à 7 soit le nombre de cartes par joueur, on continue à rejouer un tour pour chacun des participants.

Le controlFlow : sentEventDeconnexion permet de fixer la condition selon laquelle il reste des joueurs connectés ou non pour déterminer si l'on quitte la partie ou non .Si tous les clients sont déconnectés, la partie peut être quittée.

IV. Use cases

Dans cette partie on va parler des uses cases pour représenter les fonctionnalités de notre projet.

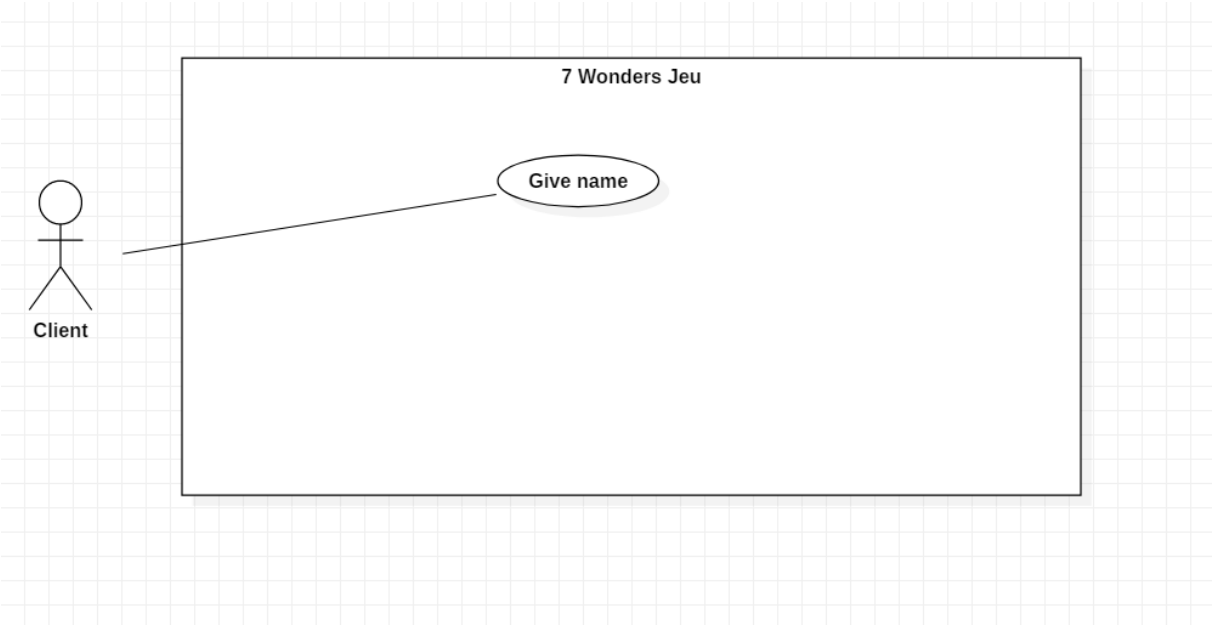
Notre projet dispose de 5 fonctionnalités principales et qui sont les axes majeurs du jeu qui sont :

1. Identification
2. Distribution des cartes
3. Distribution des merveilles
4. Distribution des pièces
5. Jouer un coup

Chaque fonctionnalité est représentée par un schéma appelé Use-Case qui sert à expliquer de le plus clairement possible le déroulement de la fonctionnalité et de ce fait le déroulement du jeu 7wonders.

1. Identification

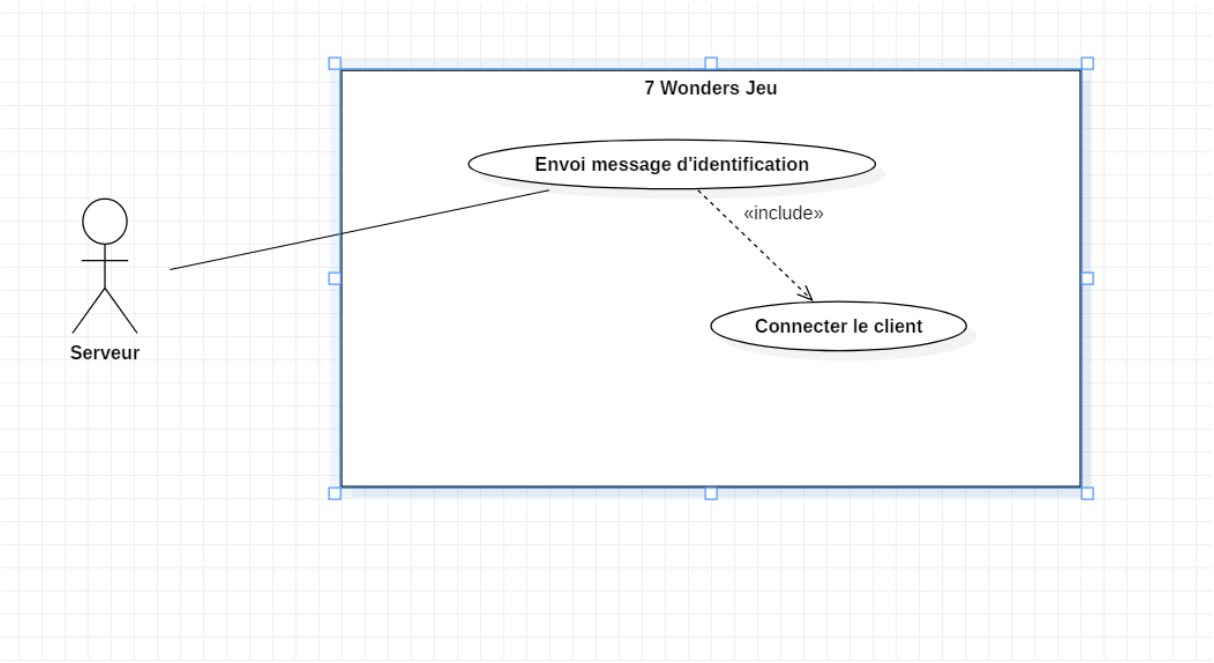
Identification coté serveur



Dans cette étape le client :

- Indique son nom dans le terminal du jeu pour se connecter

Identification coté client



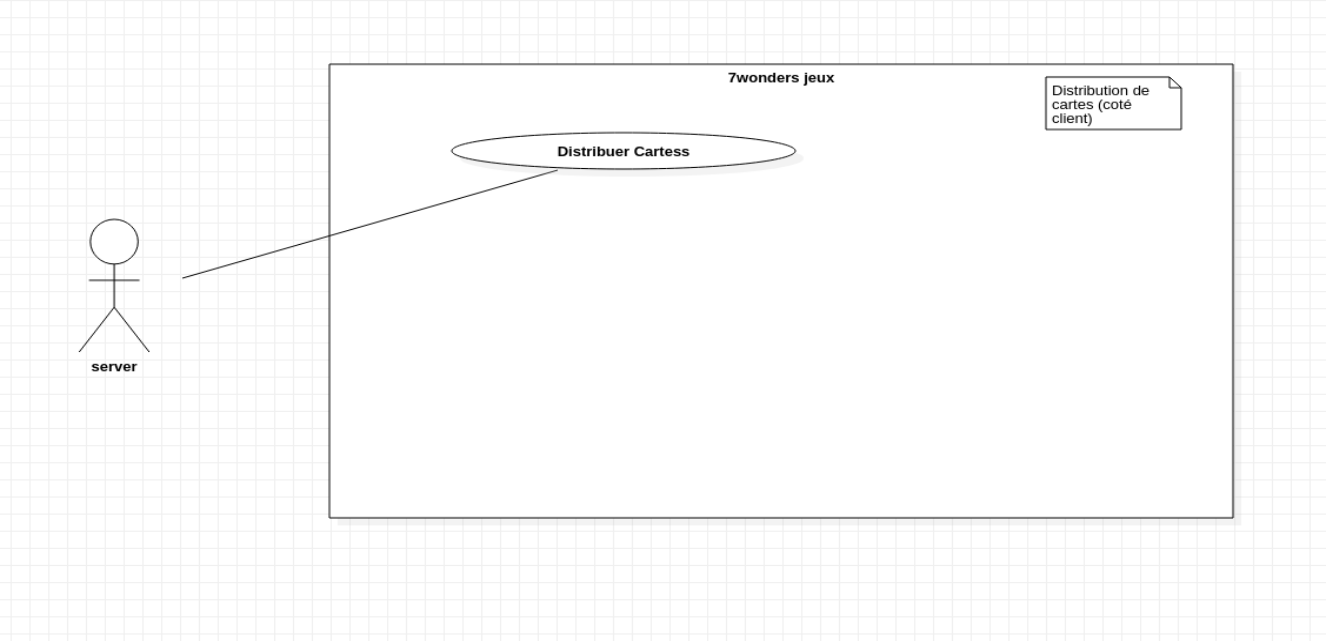
Du coté client le serveur se charge :

- D'envoyer un message indiquant la connexion du client après avoir procéder à l'étape d'identification de ce dernier qui se fait de manière automatique via un listener du coté serveur.

2. Distribution de cartes

Cette fonctionnalité permet la distribution des cartes à chaque joueur d'une manière complétement aléatoire

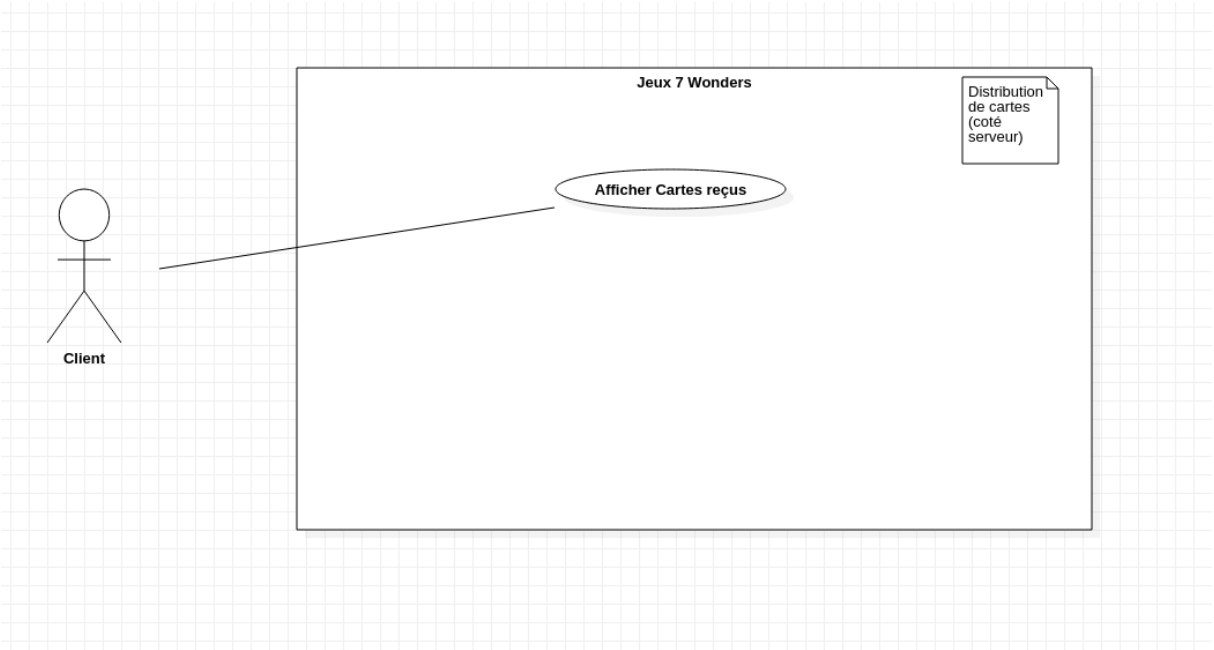
Distribution des cartes coté client



Le serveur :

Procède à la distribution des cartes pour les joueurs

Distribution des cartes coté serveur



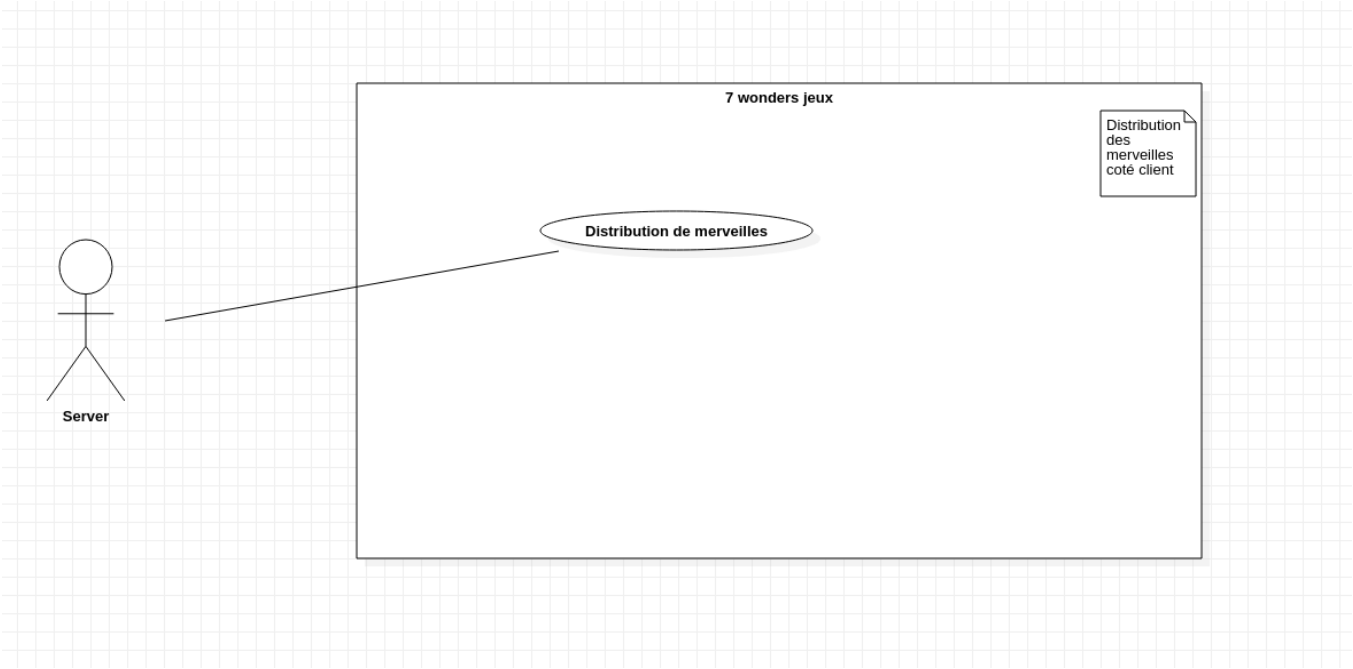
Le client :

- Affiche et met en évidence les cartes reçus

3. Distribution des merveilles

Cette fonctionnalité permet la distribution des merveilles à chaque joueur d'une manière complétement aléatoire, chaque joueur reçoit une merveille.

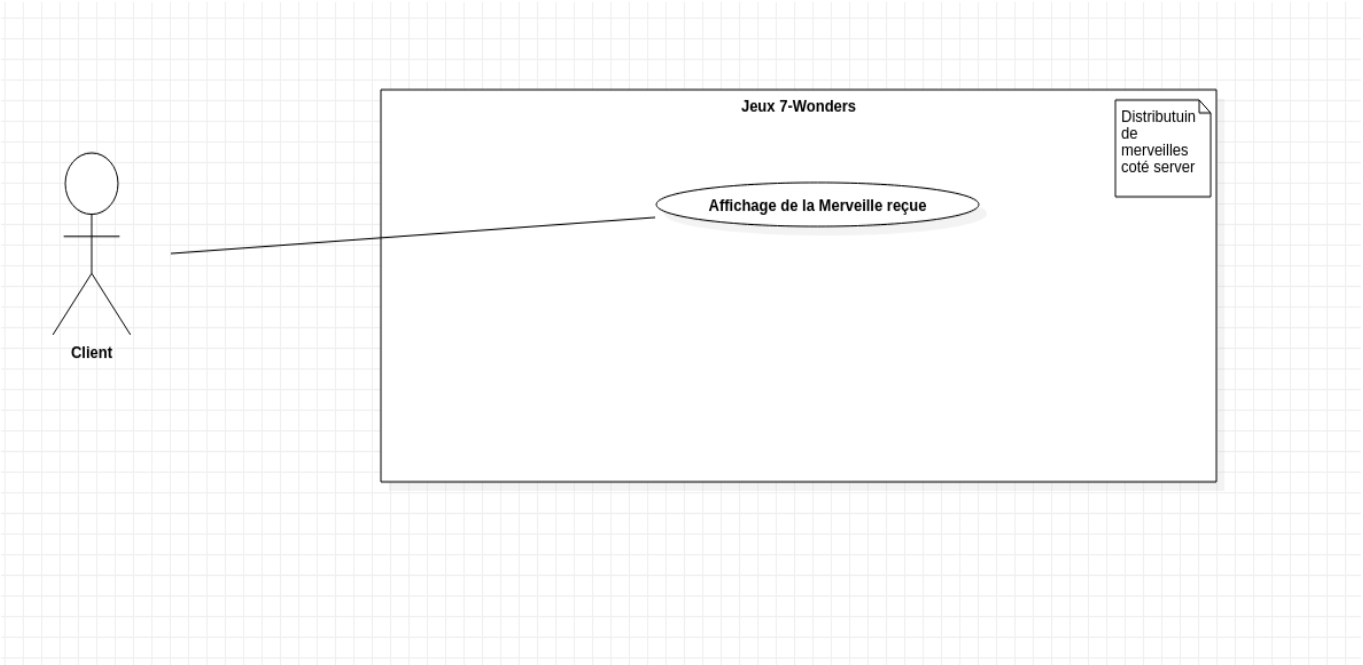
Distribution des merveilles coté client



Le serveur :

- Distribue aux joueurs les merveilles aléatoirement

Distribution des merveilles coté serveur



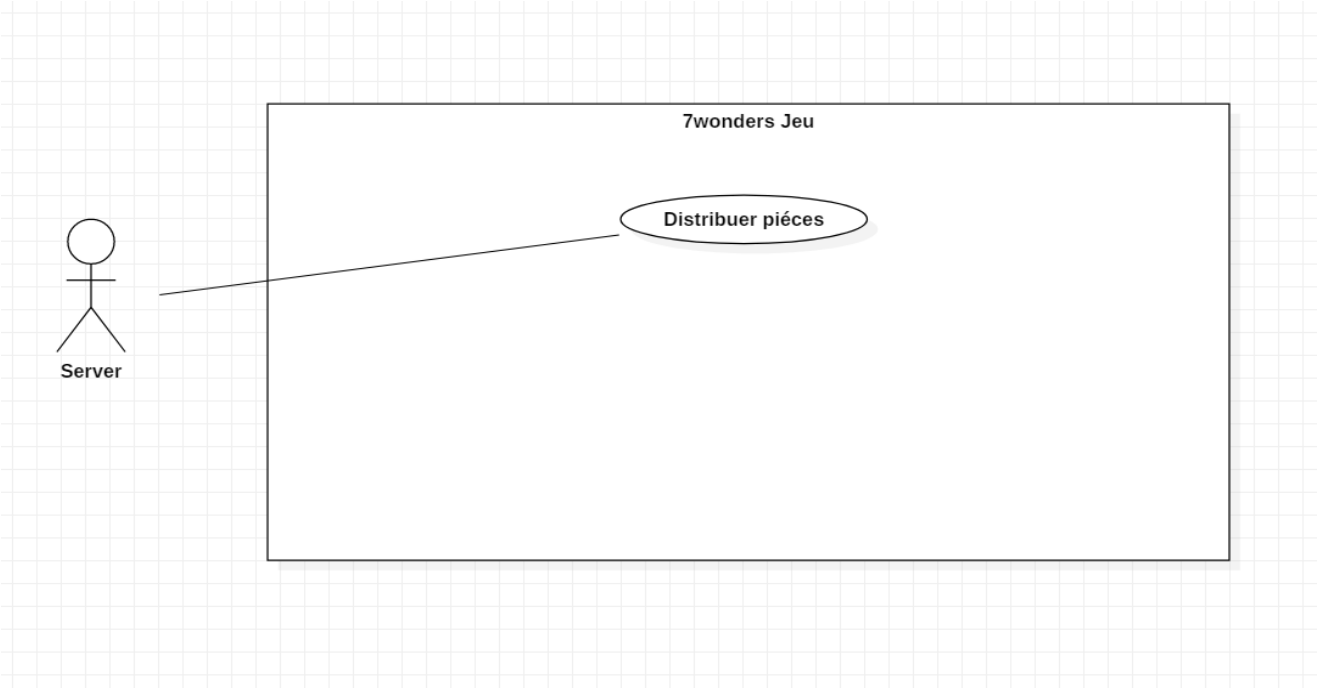
Le client :

- Met en évidence la merveille qu'il a reçu par le serveur

4. Distribution des pièces.

Dans cette fonctionnalité chaque joueur doit percevoir 3 pièces de valeur 1 de manière automatique et dès le lancement du jeu, vu que chacun dois systématiquement les posséder pour pouvoir débuter.

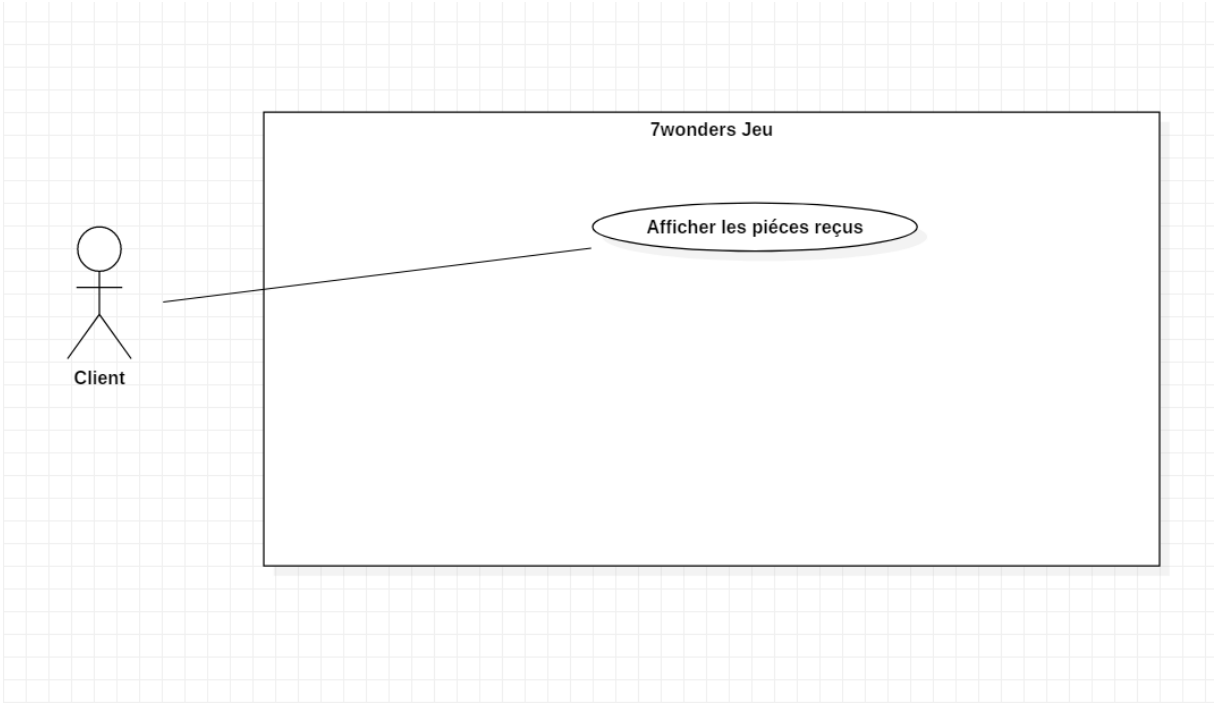
Distribution des pièces coté client



Le serveur :

- Distribue les pièces pour les joueurs.

Distribution des pièces coté serveur

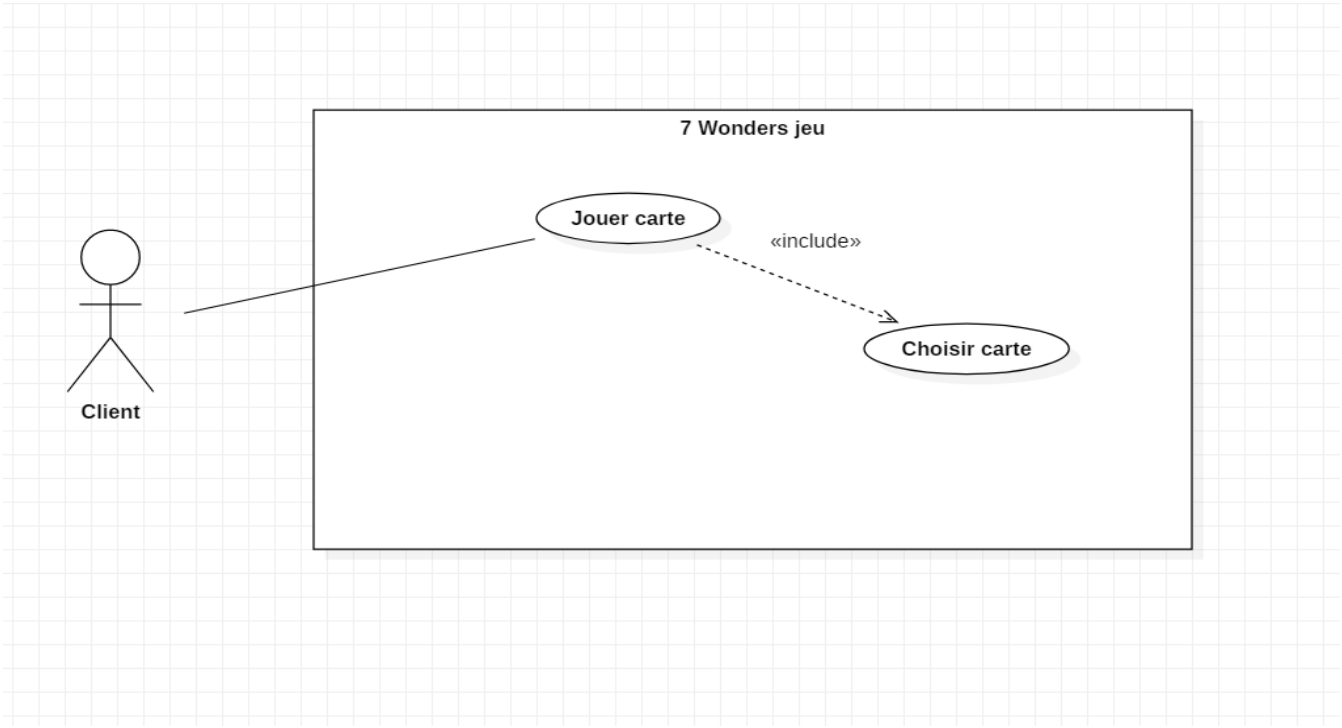


Le client : affiche que les pièces ont bien été reçus.

5. Jouer un coup

Cette fonctionnalité est la fonction principale du jeu car c'est elle qui permet de jouer tous les coups, le joueur choisis une carte et la joue et chaque carte joué diminue ces cartes restantes.

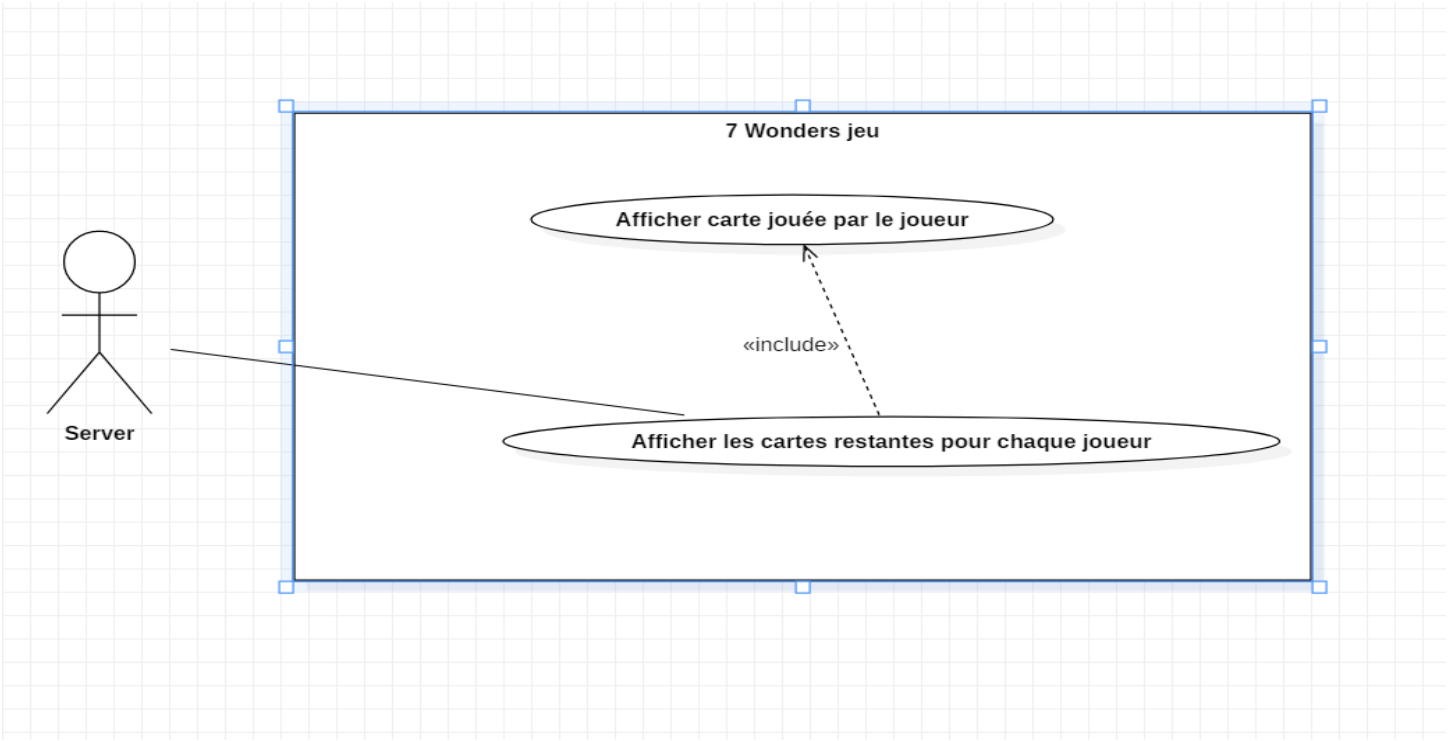
Jouer un coup coté serveur :



Le joueur :

- Procède à la sélection de la carte souhaitée.
- Joue la carte préalablement choisie.

Jouer un coup coté client

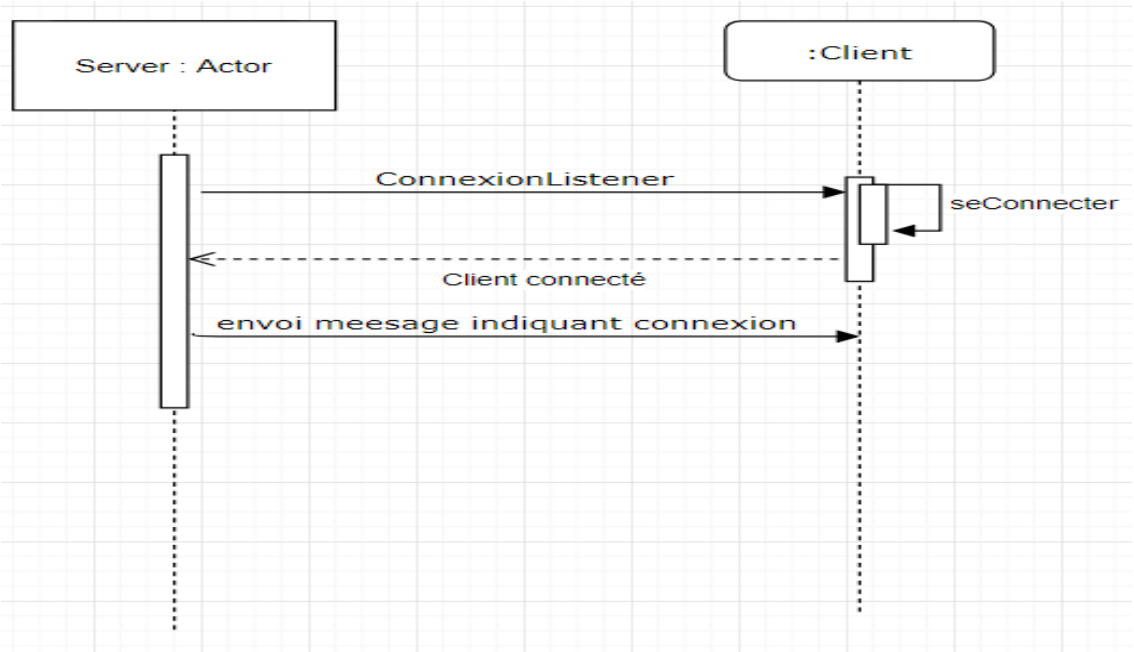


Le serveur :

- Affiche la carte jouée par le joueur
- Affiche ce qui reste au joueur après avoir jouer son coup

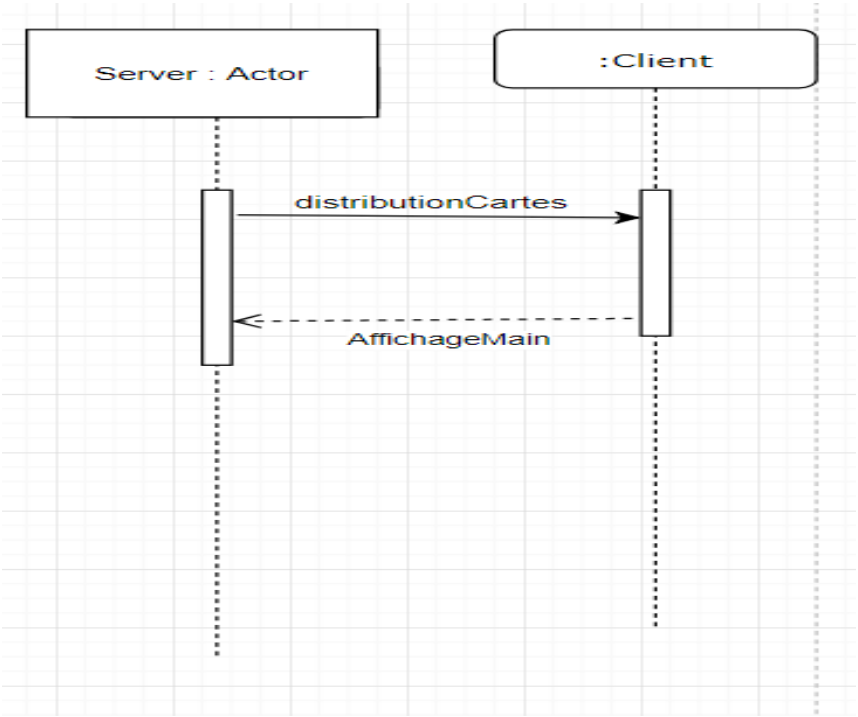
V. Diagrammes de séquence

1. Identification des clients



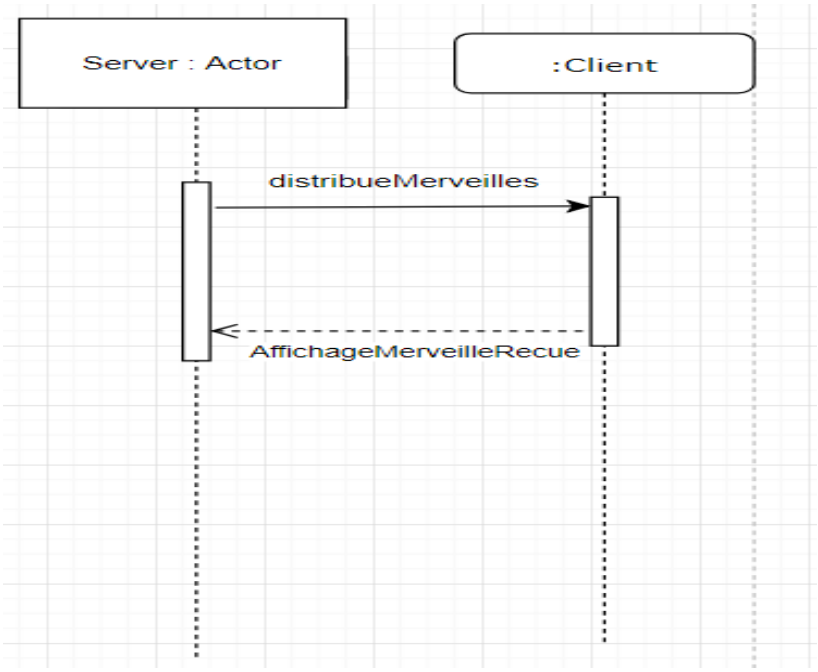
Ce diagramme illustre la connexion client-serveur. Le client initialise une partie et s'abonne aux connexion des clients afin de mémoriser leur adresse. Une fois que le client se connecte , le serveur envoie le message indiquant la connexion

2. Distribution des cartes et réception de la main



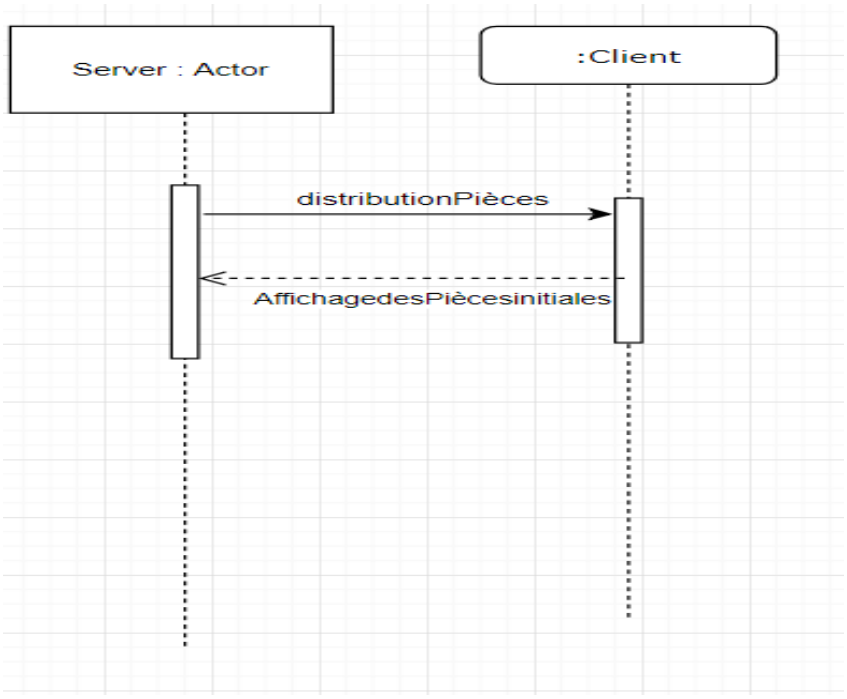
Ce diagramme de séquence modélise la distribution des cartes et leur réception par les clients. Le serveur initialise les cartes coté client et en distribue aléatoirement 7 à chaque clients. Le client affiche ensuite sa main.

3. Distribution des merveilles



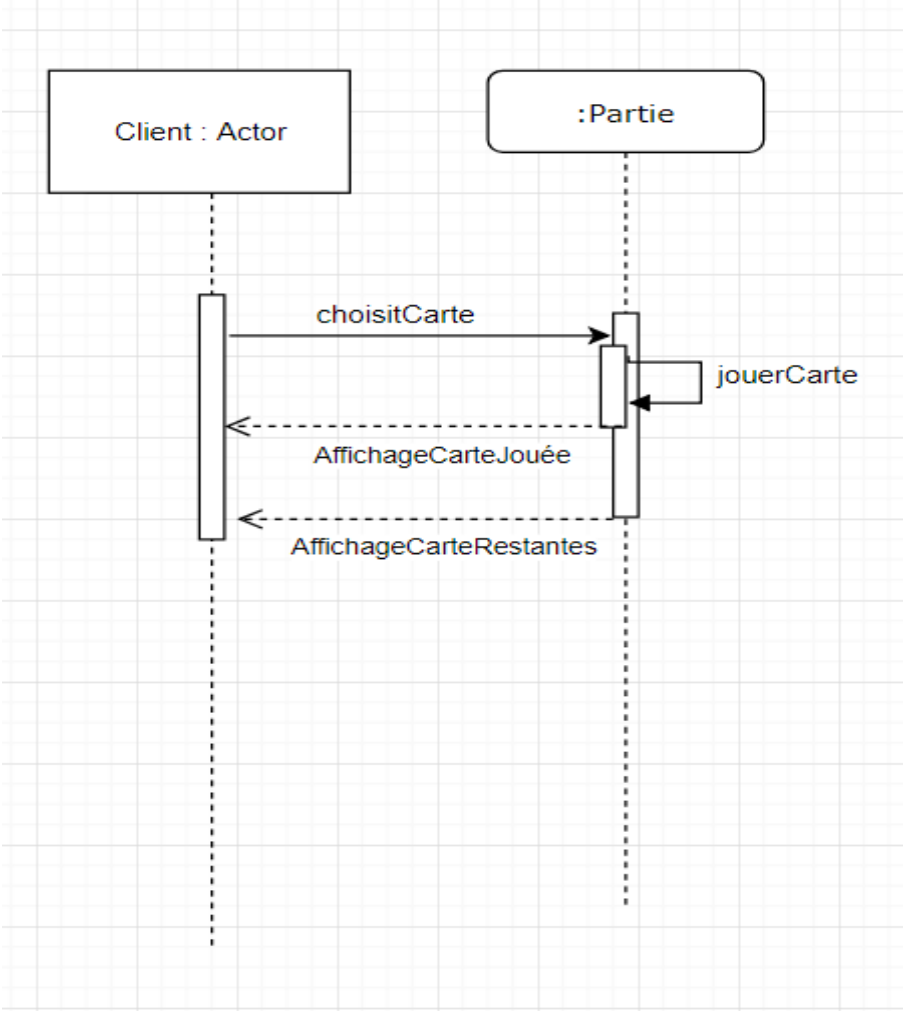
Distribution des merveilles : De même pour les merveilles, le serveur initialise les carte coté client et en distribue aléatoirement 7 à chaque client qui affiche ensuite sa main .

4. Distribution des pièces



Distribution des pièces : Ici, le serveur distribue coté client et à chaque joueur 3 pièces de valeur 1. Les clients affichent ensuite leurs pièces initiales cotées serveur.

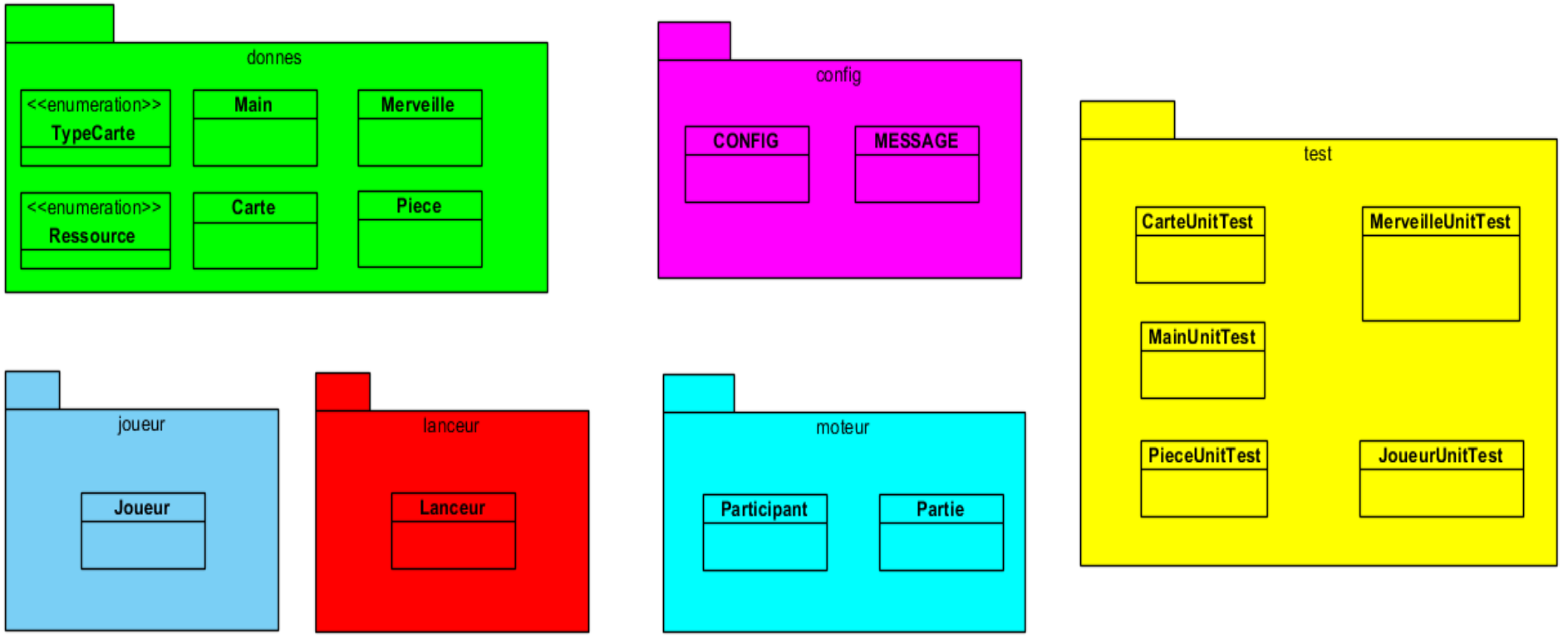
5. Jouer un tour



Jouer un tour : Au moment de jouer son tour , le client choisit une carte et la joue coté serveur. Le serveur renvoie coté client la carte jouée et affiche le reste des cartes à jouer.

VI. Diagramme de classe

1. Les packages



Le package données:

Ce package contient tous les éléments de jeu qui seront utilisés dans les autres packages pour lancer une fonction et la faire fonctionner.

Le package config :

Ce package contient des informations qui sont en static final, par ailleurs ils restent inchangés (la configuration du serveur ainsi que les messages à afficher dans la console).

Le package joueur :

Ce package contient le joueur avec tous ses caractéristiques (la merveille, les cartes, les pièces..) et il s'occupe de la partie réseau d'un client.

Le package moteur:

Ce package gère la partie réseau du serveur qui est lancé à partir de la classe Partie avec l'aide des informations du client qui se trouve dans Participant.

C'est le principal package qui lance les échanges Serveur-Client.

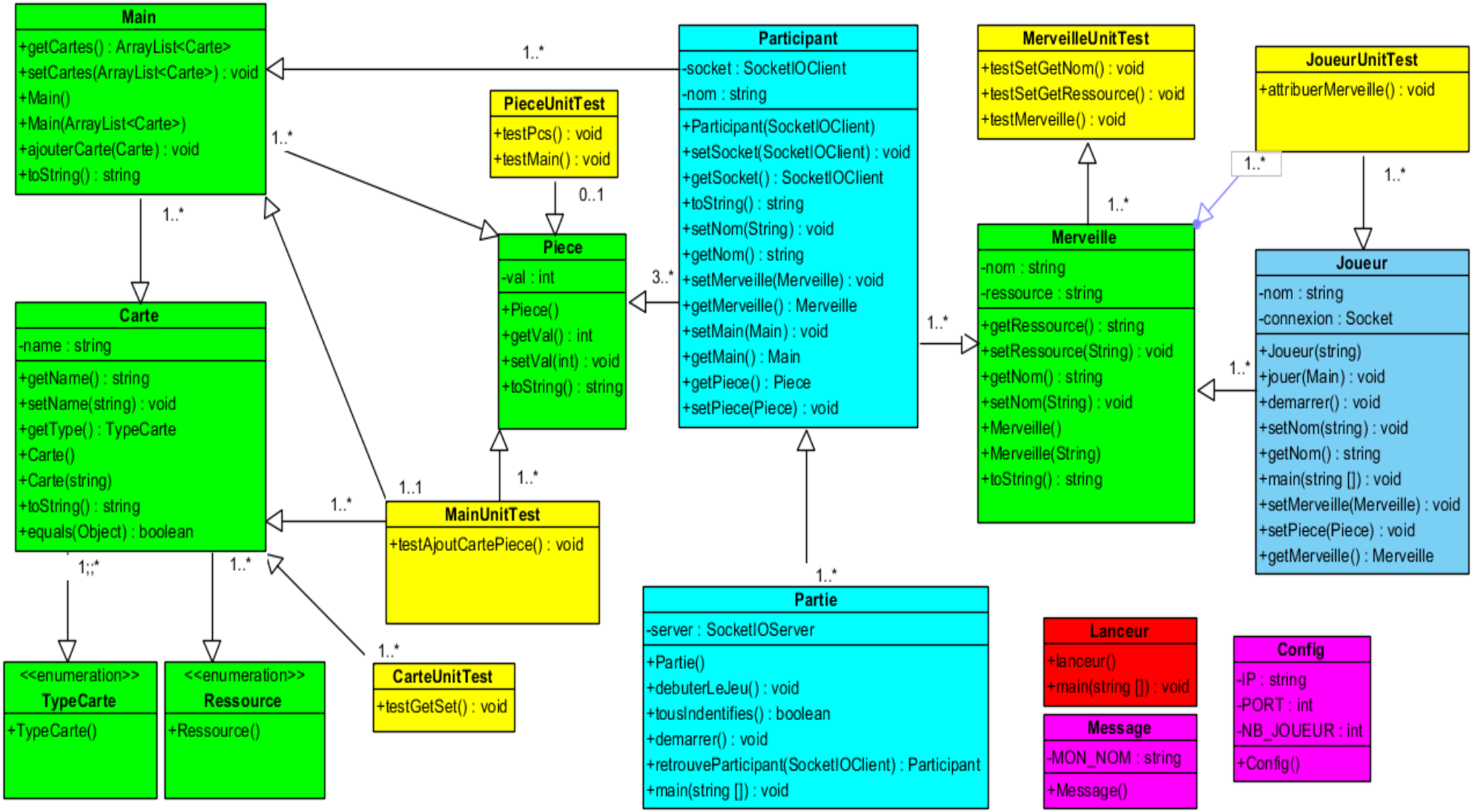
Le package lanceur:

C'est le package qui contient l'exécutable qui lance l'application.

Le package test:

C'est le package qui contient les tests des fonctionnalités des classe

2. Diagramme de classe détaillé



La partie verte : elle contient les classes (Merveille, Carte et Pièce) et les énumérations (Typecarte et Ressources) qui composent le package ‘donnees’ qui contient lui-même tous les éléments de jeu qui seront utilisés dans les autres packages surtout pour la classe Joueur.

La partie bleue foncée : elle contient que la classe Joueur qui dispose de Merveille et Piece et le socket connexion et elle sert à identifier un joueur après la connexion au serveur et lance la distribution de merveilles, des pièces et des cartes en utilisant du Json JsonObject et JsonArray.

La partie bleue claire : cette partie gère les échanges entre Client et Serveur, la classe Participant contienne les informations du client et s’occupe de la partie client alors que la classe Partie lance le coté Server.

La partie violette : elle contienne deux classe pour le stockage des informations non changeable ; la classe Config pour les données de config et la classe Massages pour les données affichées sur la console

La partie rouge : c’est la partie principale qui lance l’application et contienne l’exécutable.

La partie jaune : c’est le package test qui est créer en parallèle avec le dossier main de chaque module

VII. Conclusion

1. Analyse de notre solution (points forts et points faibles)

Points faibles :

Retard considérable pour le lancement de la production du projet pour des causes personnelles qui ont fortement impacté sur ce dernier.

Problèmes techniques entre la compréhension du MAVEN en synchronisation Avec GitHub.

Difficulté de compréhension du client serveur à cause de gros problèmes personnels qui a résulté d'absences lors des séances des cours du client serveur.

Problèmes de modélisation pour certains points du déroulement du jeu

Points fort :

Evolution lors des dernières itérations grâce aux conseils et remarques de Mme. CABRIO.

2. Evolution prévue

Mettre en place un bon nombre d'améliorations et gérer bien le temps. Nous envisageons de continuer ce projet.