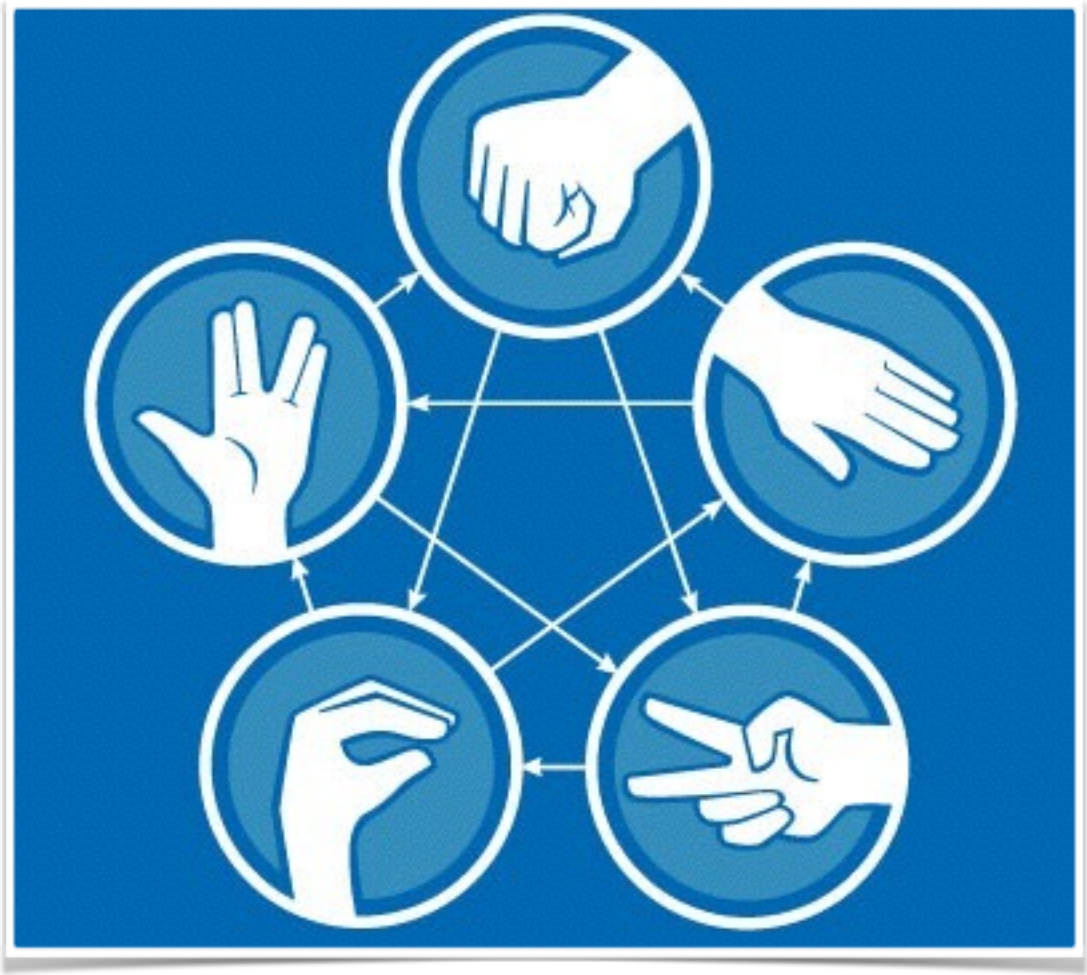
NOM DE SOCIÉTÉ



Dossier de conception
Application Clients-Serveur

Préparé pour : François PFISTER
Préparé par : Frédéric BONNAND, Loic VERGER
10 novembre 2014

Introduction

Objectif

L'objectif est de concevoir et maquetter un jeu en réseau.
Le langage utilisé sera Java, basé sur un principe de Clients - Serveur à l'aide de sockets de connexion.

Le jeu retenu a été une version moderne du ShiFouMi adapté de la série télévisée « Big Bang Theory ».

Sommaire

RÉDACTION DU BESOIN

Le Besoin

Deux joueurs doivent pouvoir se connecter et s'affronter autours d'une partie de « Pierre Feuille Ciseaux Lézard Spock » en 3 duels gagnants.

Qu'est ce que le jeu « Pierre Feuille Ciseaux Lézard Spock (PFCLS)» ?

Il s'agit d'une variante du traditionnel ShiFouMi (ou Pierre Feuille Ciseaux) inventé par Sam Kass and Karen Bryla mais rendu célèbre par le personnage de Sheldon dans la série « Big Bang Théory ». Cette version moderne du jeu a pour intérêt d'augmenter les combinaisons possibles et de diminuer le nombre de duels « nuls » au cours d'une partie*. (Cf la section « Manuel d'utilisation » pour lire les règles)*

EXIGENCES FONCTIONNELLES

Ce jeu en réseau de type Clients-Serveur est une version moderne du Shi Fou Mi, voici les exigences fonctionnelles liées à cette application :

- L'application permettra la connexion simultanée de plusieurs personnes sur le serveur
- Lorsque deux personnes sont connectées, elles doivent être mises en relation afin de s'affronter
- Le joueur pourra choisir un signe parmi cinq
- Le joueur pourra choisir un pseudo
- Le joueur pourra adapter sa fenêtre de jeu

EXIGENCES TECHNIQUES

- L'application doit pouvoir créer un nouveau serveur de jeu quand le nombre de joueurs connectés est impair
- L'application permettra la connexion simultanée de plusieurs personnes sur le serveur
- Lorsque deux personnes sont connectées, elles doivent être mises en relation afin de s'affronter
- Lors du lancement du client, l'application doit démarrer en moins de 5 secondes.
- L'application doit fournir une interface réglable
- L'application devra traiter les choix des joueurs
- L'application devra considérer les pseudos des joueurs
- L'application devra déterminer un vainqueur pour chaque manche
- L'application devra déterminer un vainqueur final

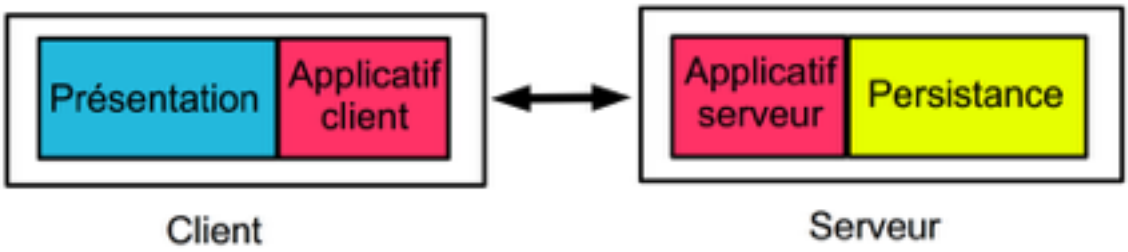
ARCHITECTURE DU SYSTÈME

La structure du réseau

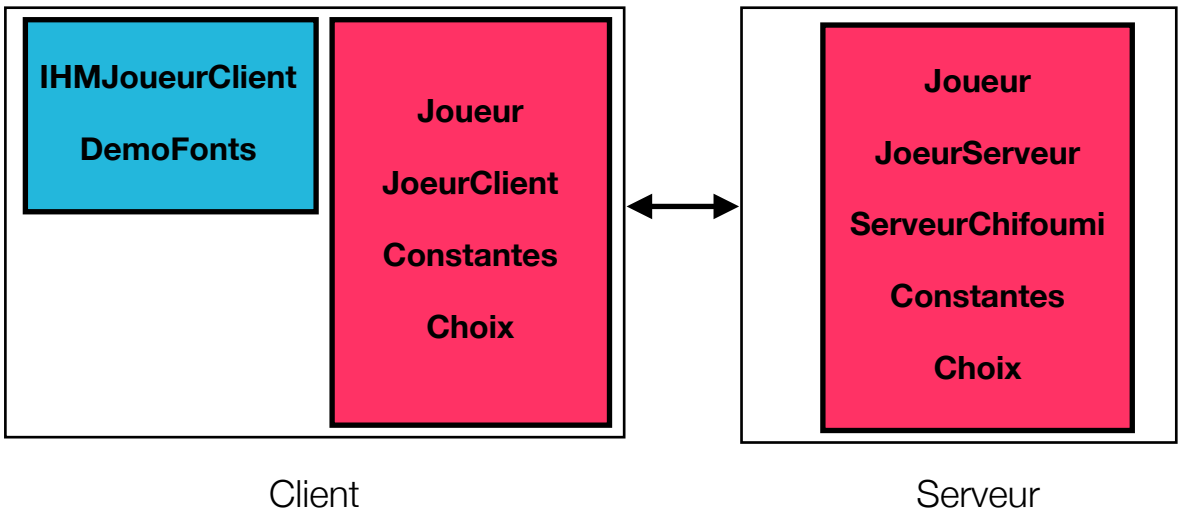
L'application repose sur une architecture 2-tiers.

Client / serveur de base, avec 2 éléments :
Client : présentation, interface utilisateur
Serveur : partie persistance, gestion physique des données

Les services métiers/partie applicative sont découpé entre la partie serveur et le client.



Dans le cas présent :



ARCHITECTURE DU SYSTÈME

La structure de l'application

L'application se décompose en deux parties distinctes et donc en deux applications Java différentes.
Une application sera lancée sur le Serveur, l'autre chez le client qui souhaite utiliser l'application.

La JavaDoc est fournie en annexe de ce dossier.

1 - Côté Serveur

L'application coté Serveur comprend 5 classes :

- Constates.Java
- Choix.Java
- Joueur.java
- JoueurServeur.java
- ServeurChifoumi.Java

Constantes.java et Choix.Java ne sont que des fichiers de constantes. Le premier contient les différentes choix possibles, et le deuxième les différentes armes disponibles.

N.B. : Ces deux fichiers sont présents à l'identique dans l'application côté client.

```
package cs;


public class Constantes {
         public static final int POINT = 1;
         public static final int JOUER = 2;
         public static final int NUM = 3;
         public static final int CHOIX = 4;
         public static final int GAGNE = 5;
         public static final int PERDU = 6;
         public static final int EGALITE = 7;
         public static final int PARTIE_GAGNEE = 8;
         public static final int PARTIE_PERDUE = 9;
         public static final int CHOIX_ADVERSAIRE =
11;

         public static final int CIBLE = 3;
}
```

Constantes.java

```
package cs;

public enum Choix {
    CAILLOU,  CISEAUX, FEUILLE, LEZARD, SPOCK;
}
```

Choix.java

La classe Joueur.java

Cette classe est une classe qui implémente une interface Runnable. Elle définit un joueur par une socket et contient les méthodes de bases pour déterminer si le joueur a gagné, perdu ou fais match nul au denier duel.

Elle contient également une méthode permettant de savoir quand la partie est terminée (que l'un des deux joueurs a atteint le nombre de victoires requises).

Méthodes :
aGagne();
aPerdu();
egalité();
partieGagnee();
le run() du Thread

La classe JoueurServeur.java

Cette classe hérite de la classe Joueur précédemment étudiée. C'est une classe adaptée avec des méthodes spécifiques nécessaire pour un joueur côté serveur.

Méthodes :
aGagne();
aPerdu();
egalité();
envoyerNumero();
traiterMessage();
le run() du Thread

La méthode run du Thread a pour mission de lire les messages envoyer par les clients tant que la partie n'est pas déclarée comme terminée.

traiterMessage(); a pour rôle de traiter les requêtes venant du client, soit le signe que le client choisi pour chaque duel. Le signe choisi par un joueur est alors envoyé à son adversaire pour l'informer et une autre méthode vérifie quel joueur a gagné le duel. Cette méthode examinerChoix() est une méthode de la dernière classe présente sur le serveur, la classe ServeurChifoumi.java.

La classe ServeurChiFouMi.java

Cette classe classe est la classe exécutée au lancement du serveur. elle contient la méthode main().
A son lancement un serveursocket est créé, et on attend l'arrivée de deux demandes de connexion provenant de deux joueurs différemment avant le « lancer le feu vert » pour le début de la partie, grâce à la méthode donnerFeuVert();

2 - Côté Client

La classe Joueur.java

Elle est identique à la classe Joueur.java présente sur le serveur. Elle assure la cohérence de l'Objet Jouer entre le client et le serveur.

La classe JoueurClient.java

Cette classe classe est la classe exécutée au lancement du Client. Elle étend la notion de joueur avec des méthodes spécifique au joueur côté client.
Un ClientServeur est définit par son attribut socket qui le lie au serveur.

```
public JoueurClient(Socket socket)  throws Exception{
        super(socket);
        adversaire = new JoueurClient();
}
```

Au lancement de l'application, cette classe demande le pseudo du Joueur puis lance l'interface IHMJoueurClient qui est le plateau de jeu.

La classe comprend 3 méthodes :
La méthode Main, la méthode run() du Thread et la méthode responsable de traiter les messages venant du serveur : traiterMessages().

Les classes Constantes.java et Choix.java

Ces classes sont rigoureusement identiques à celles qui sont placées sur le serveur. Elle permette d'assurer la cohésion entre le client et le serveur.

La Classe IHMJoueurClient

Cette classe est la vue qui représente le plateau de jeu pour le client.
Il lui permet de choir son arme et de voir l'arme choisie par son adversaire.
Le tableau de jeu permet également de voir son score et celui de son adversaire.

- Plateau de jeu du client -

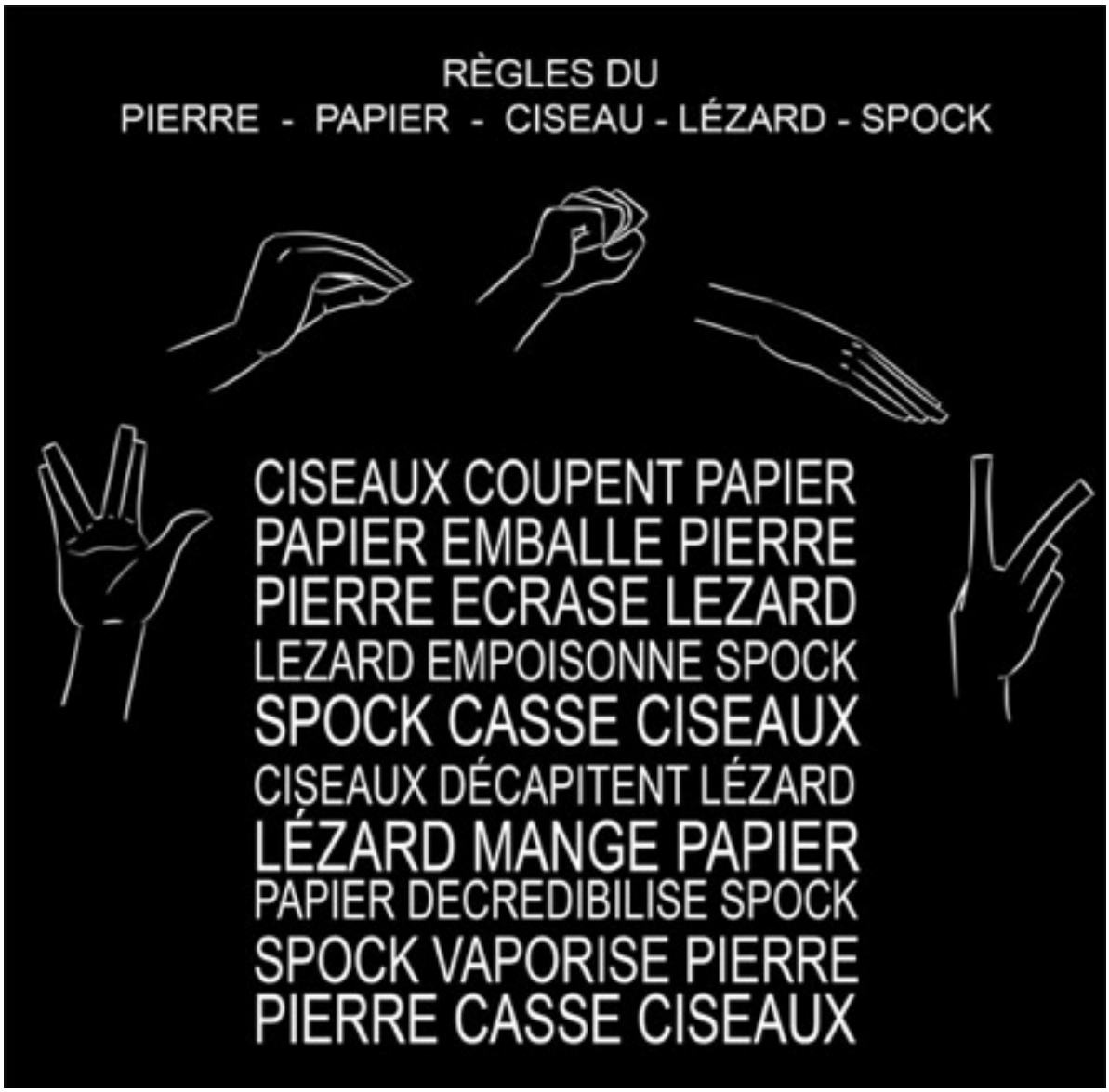
CRITÈRE POUR LA QUALITÉ

Pour être reconnue qualitativement, l'application devra remplir les critères suivant :

- Facilité d'utilisation (design clair et attractif, application ergonomique, application facile à comprendre)
- Application répondant aux besoins exprimés (exigences ..)
- Portabilité afin de pouvoir l'utiliser n'importe où (adaptation et installation facile)
- Performance (temps de réponse, utilisation des ressources, etc.)
- Code commenté et maintenable pour permettre à d'autres personnes d'analyser le code mais aussi de l'adapter à leur guise.
- Fiabilité (bugs ..).

MANUEL D'UTILISATION

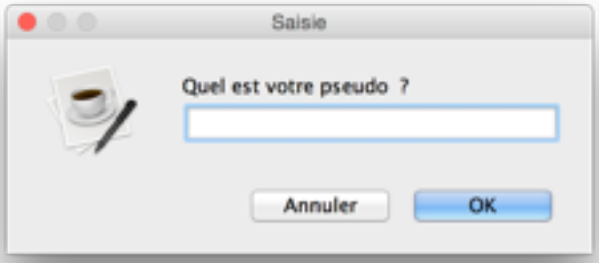
Règles du jeu



Didacticiel du jeu

1- Lancement de l'application

2- Renseigner le pseudo



3- Attendre la connexion du second joueur

4 - Une fois les deux joueurs connecter, choisir l'arme utilisé pour le duel

4 - Une fois que les deux joueurs ont choisis leur arme, affichage des résultats

5 - Les scores sont mis à jour, un nouveau duel peut être commencé (étape 4).

6 - Le premier joueur arrivé à 3 points gagne la partie

CONCLUSION

Voici comment est conçue cette application de ChiFouMi revisité par des fans de Star Trek.

Cette application est un cas assez trivial d'un jeu multijoueur Client-Serveur. Pour aller plus loin, nous pourrions penser à des implémentations de fonctionnalités qui verraient le jour dans la prochaine version du jeu comme :

- Choisir le nombre de points gagnants pour remporter la partie.
- Pouvoir proposer à son adversaire une revanche.
- Possibilité de choisir son adversaire si plus de deux joueurs sont connectés.
- Implémenter un chat en parallèle du jeu.