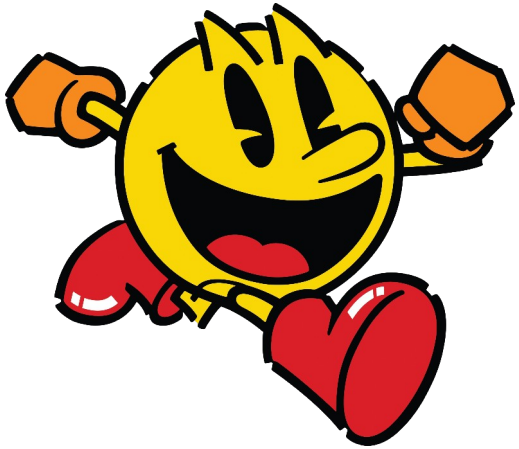
PAC-MAN™



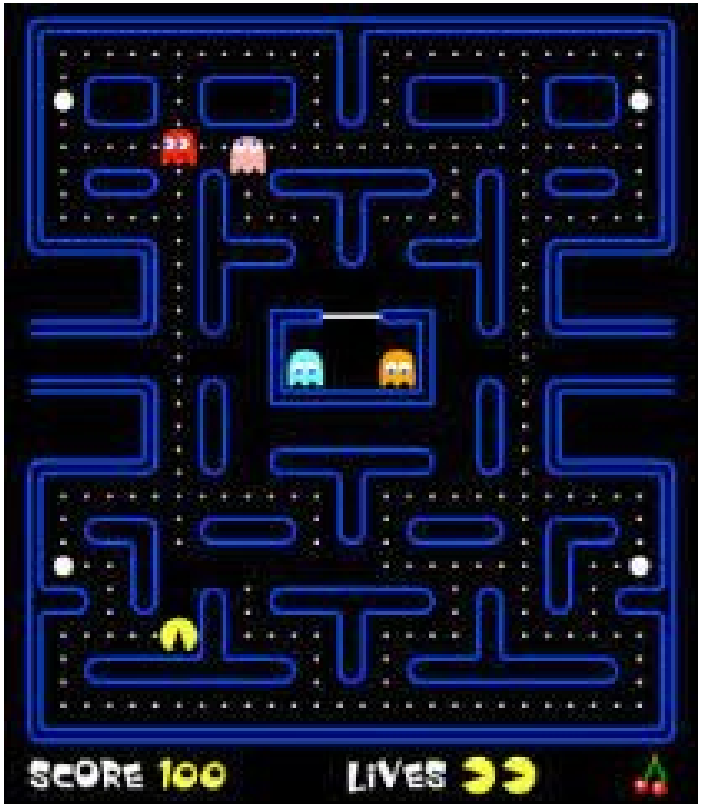
PAC-MAN™

- Présentation de notre choix de jeu et du rôle du serveur dans notre système
- Cahier des charges du jeu et du système serveur-client
- Les outils exploités
- La conception de notre système avec le diagramme de séquence
- Les tâches que nous nous sommes réparties
- Le rendue finale du jeu
- Bilan

PAC-MAN™

Nous avons décidé de faire un jeu Pac-Man qui serait entièrement hébergé par le serveur. Ainsi, le client n’a qu’a se connecter au serveur pour jouer.

Dans Pac-Man, Pac-Man qui est le joueur, doit manger des “gommes”, des petites billes jaunes. S’il est touché par les fantômes, il meurt. Au bout de 50 points marqués, il gagne.

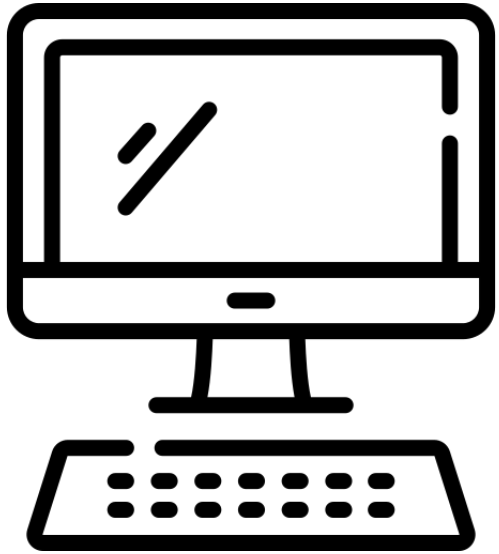


Cahier des charges

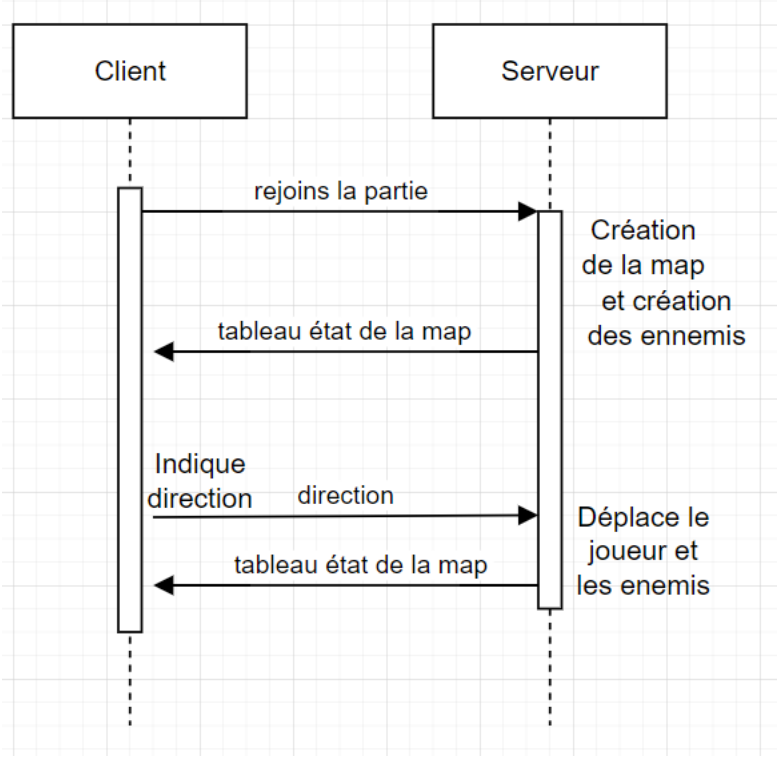
- Déplacement du pacman de haut en bas et de droite et gauche
- Gestion des collisions avec les bords et les murs :
    - ne peut pas traverser les bords et les murs
- Gestion des fantômes et de leurs déplacement :
    - augmentation de la difficulté :
        distance de déplacement du fantôme réglable et leur « intelligence a chercher le Pac-Man» aussi.
- Map qui s'affiche côté serveur et côté client de manière synchronisée
- Le serveur héberge le jeu
- Le client joue le pacman
- Les points marqués se font au déplacement du joueur,
  Le joueur ne peut pas avoir de points en restant sur place.

Outils utilisés

- Visual Studio Code
- Raspberry Pi
- Bibliothèques keyboard et msvcrt pour gérer les entrées claviers



Conception (diagramme de séquence)



Le modèle est un modèle serveur-client.

Le client se connecte au serveur.

Le client envoie des messages au serveur pour indiquer la direction, qui va déclencher le déplacement du joueur sur la map. Le client peut aussi arreter sa connexion au serveur en appuyant sur la touche « c ».

Le serveur à chaque modification apportée va faire une mise à jour de la map et va la renvoyer au client.

Réalisation - Justine

- Gestion des connexions réseaux
- Affichage de la map du serveur côté client
- Mise à jour de la map du serveur en fonction des instructions envoyé par le client
- Mise à jour de la map du serveur pour l'affichage client (synchronisation entre la map serveur et la map serveur affiché côté client)
- Création du système de score du jeu et de victoire du joueur

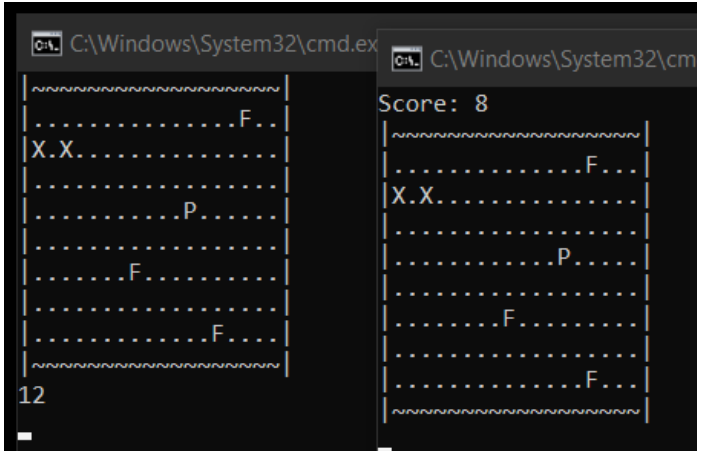
Réalisation - Clément

- Création de l’esquisse de jeu
    - Création des classes de la map pacman et des ennemis fantômes
- Gestion des murs
- Ajout des input clients
- Création du système de difficulté intelligente

Démonstration (captures d'écrans)

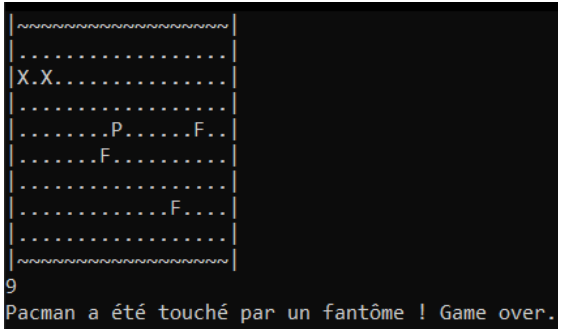
Le client écrit *start* sur le terminale pour lancer le jeu

Affichage de la map synchroniser
d'un côté la map *côté serveur* de l'autre
l'affichage de la map seveur *côté client* :

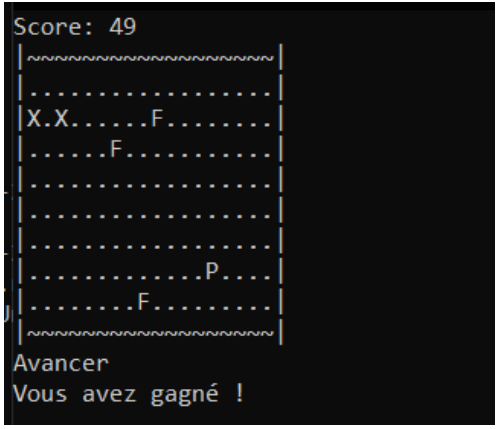


Exemple au cour d'une partie :

Quand le joueur perd…



Quand le joueur gagne !



Conclusion

Nous avons rencontré des difficultés pour la réalisation d’un affichage lisible et fluide lorsqu’il ce mets à jour côté client.

Finalement, nous avons réussis à développer le jeu côté serveur et à faire jouer un client au jeu, en lui affichant la map sur laquelle il joue. Un affichage de la map qui est synchronisé et fluide avec les modifications apportées par le jeu.

Le programme est une implémentation d'un jeu Pac-Man en mode client-serveur, utilisant des sockets UDP pour la communication entre le client et le serveur. Le client contrôle le personnage Pac-Man à l'aide des touches 'z', 'q', 's' et 'd', tandis que le serveur gère la logique du jeu, la mise à jour de la carte et la gestion des fantômes.

Durant ce projet, nous avons appris a gérer les messages entre client et serveur et comment les utiliser pour réaliser notre jeu.