# ENSTA Paris

École Nationale Supérieure de Techniques Avancées

**Rapport du projet IN204**

**TetrisSS**

Kevin Alexandro SANCHEZ DIAZ
Yevhenii Sielskyi

**13 février 2020**

# Table des matières

# 1 Introduction

## 1.1 But du projet

Ce projet réside dans l'implémentation complète du jeu arcade TETRIS et d'étendre ce jeu en implantant la possibilité d'un jeu multijoueurs.

## 1.2 Tâches demandées

Il est nécessaire de réaliser une analyse de la faisabilité du logiciel, proposer une architecture de ce logiciel et ensuite implanter un démonstrateur reposant sur cette analyse.

1. Analysez le problème, sachant qu'il est souhaitable d'avoir une solution évolutive, permettant d'ajouter au fur et à mesure des fonctionnalités au logiciel, notamment progressivement étendre les règles du jeu. Il est notamment demandé de faire :
   (a) Une analyse d'usage du logiciel (use case analysis).
   (b) Une analyse fonctionnelle du logiciel.
   (c) Une architecture gros grains du logiciel.
2. Réalisez un démonstrateur pour valider l'approche et les choix techniques et s'assurer que ces derniers sont pertinents. Dans le cas d'une approche évolutive, il sera possible tout d'abord d'implanter un noyau applicatif (nombre restreint de pièces, mouvement limités aux déplacer latéraux ou aux rotations dans un sens seulement) et ensuite d'ajouter les éléments supplémentaires (pièces supplémentaires, calcul du score, gestion des joueurs).

Il est notamment demandé de faire :

1. Une description du modèle d'architecture.
2. De bien documenter les choix des classes.
3. De réaliser une implantation documentée du logiciel.

L'outil devra fournir au minimum les services suivants :

1. Implanter complètement les règles du Jeu TETRIS simple utilisateur.
2. Soit proposer un mode permettant de jouer avec deux ou plus de joueurs.
   (a) Chaque joueur joue sur un ordinateur différent.
   (b) Chaque joueur peut inviter les autres joueurs à une partie commune.
   (c) Il est possible d'avoir l'ordinateur qui simule un joueur.
3. Soit au contraire de proposer une version 3D de TETRIS à l'instar de BLOCKOUT.

# 2 Description fonctionnelle

L'équipe des développeurs de TetrisSS a décidé d'implémenter la version du jeu TETRIS ayant la fonction de multijoueur où chaque joueur est capable soit de créer une nouvelle session soit de trouver des sessions déjà prêtes pour y jouer en tant qu'un client.

## 2.1 Chronologie du développement

Le fonctionnel du projet évoluait comme suit :

1. Tout d'abord, il a été décidé de fournir la classe de **board** principal de jeu sous la forme d'une matrice 2D dont les éléments - certains codes de couleurs correspondant aux cellules de la fentre graphique à l'avenir. La classe abstraite de **figure**s a été établi en manière d'un vecteur dynamique de tuples de **point**s 2-dimensionnelles ayant des coordonnées qui correspondent aux indices de la matrice de tableau.
2. Ensuite, les figures classiques sont créées grace à l'utilisation de la technique du polymorphisme. La classe tableau de jeu obtient 2 nouveaux paramétres : figure actuelle et figure prochaine.
3. Le movement de la figure actuelle est mis en pratique en changeant au fur et à mesure des valeurs de la matrice au code de couleur de la figure selon son type et en augmantant les coordonnées de figure.
4. Les fonctions de movement à gauche et à droite, le critère de game over et la logique de calcul de score sont élaborés. L'operateur de sortie est surchargé pour pouvoir tester la prémière version du programme.
5. Puis, la conception de l'interface graphique primitif et l'architecture de menu sont établies. En fin, le GUI est mis en place à travers de l'utilisation de bibliothéque graphique SFML, la classe tableau obtient le paramètre grille - le groupe de carées SFML dont les couleurs changent selon le code de couleur de la matrice initiale.
6. La fonction de la rotation de chaque figure est réalisée en deux sens, les coordonées de chaque point de figure pendant la rotation changent au moyen de la multiplication matricielle avec la précision du point de référence, les figures sont distingué en fonction du type de sa rotation. Il en éxiste 3 types :
   (a) figures sans rotation ;
   (b) figures avec la rotation simple (2 états possibles) ;
   (c) figures avec la rotation complèxe (4 états, 2 sens).

7. Il est décidé de réunir toutes les fonctions correspondantes aux parties de menu differentes au sein d'une seule classe **menu** sachant la présence de plusieurs éléments communs de fonctions mentionées.
8. Les prémières démarches de multiplayer sont éféctuées, pour cela le protocle UDP, permettant d'envoyer des messages depuis le serveur aux plusieurs déstinations simultanément (broadcast), est utilisé (la librairie SFML fournit égalément des outils d'UDP) :
   (a) la classe **player** est créée avec tels paramètres comme UDP socket et nickname ;
   (b) la classe-fille **client** de player est créée avec des possibilités de chercher des serveus, y connecter en utilisant des ports UDP, écouter le serveur, recevoir ses commandes et envoyer des informations nécessaires pour l'affichage de jeu correcte ;
   (c) la classe-fille **server** de player est créée avec des possibilités d'écouter des clients, recevoir leurs informations, les broadcaster, gérér sa session.
9. La connexion au niveau de session menu est établie, les clients et le serveur peuvent échanger des status et les mettre à jour sur leurs écrans.
10. Les fonctions dédiées à l'écoute et à l'envoie d'information des clients et de serveur sont parallélisées avec le thread principal du programme en appliquant la technologie de threads relativement simple, fournie par SFML.
11. La connexion est finalement établie au niveau de jeu, chaque joueur est capable de mettre à jour les tableaux et les scores d'autres joueurs grace à la communication continue entre eux à travers de parallélisme.
12. La logique de l'augmentation du niveau de jeu est ajoutée (level up). Les niveaux augmentent en fonction de nombre de lignes effacées, par contre, le joueur est toujours capable de choisir l'un des niveaux initiales possibles. Il en existe 3 niveaux initiales différents :
    (a) niveau "Mechanics" - correspond à niveau 1 ;
    (b) niveau "STIC" - correspond à niveau 3 ;
    (c) niveau "Applied Maths" - correspond à niveau 5.

    Le niveau maximal d'attaindre est 30.
13. Les dérnieres modifications de GUI sont mis en place, les comportements différents du logiciel sont testés de manière empirique et des erreurs visibles sont éliminées.

## 2.2 Scalabilité du code

Grace à l'implementation de l'approche du polymorphisme dans le cadre des figures différentes et à l'utilisation de constantes partout dans code, création des propres types de données, il est possible d'ajouter des fonctionnalités supplémentaires (ajouter des nouvelles figures, changer la taille du tableau de jeu, changer la vitesse de jeu etc) assez rapidement sans affecter la conception du logiciel.

# 3 Architecture

La Figure 1 represente l'architecture des classes de l'application et toutes les liens entre eux. Les classse, dont les fonctions sont décrits dans la chapitre 2.1, sont suivantes :

1. Point - 2D point avec ses coordonnées ;
2. Figure - la figure abstraite (le groupe de Points avec son code de couleur) ;
3. Board - le tableau de jeu avec la matrice de couleurs, la grille graphique, niveau, score et 2 figures - actuelle et prochaine ;
4. Menu - les foctions différentes du menu de logiciel ;
5. Player - le joueur avec ses UDP socket et nickname ;
6. Client - un joueur-invité dans la session ;
7. Server - un joueur-propriétaire de la session.

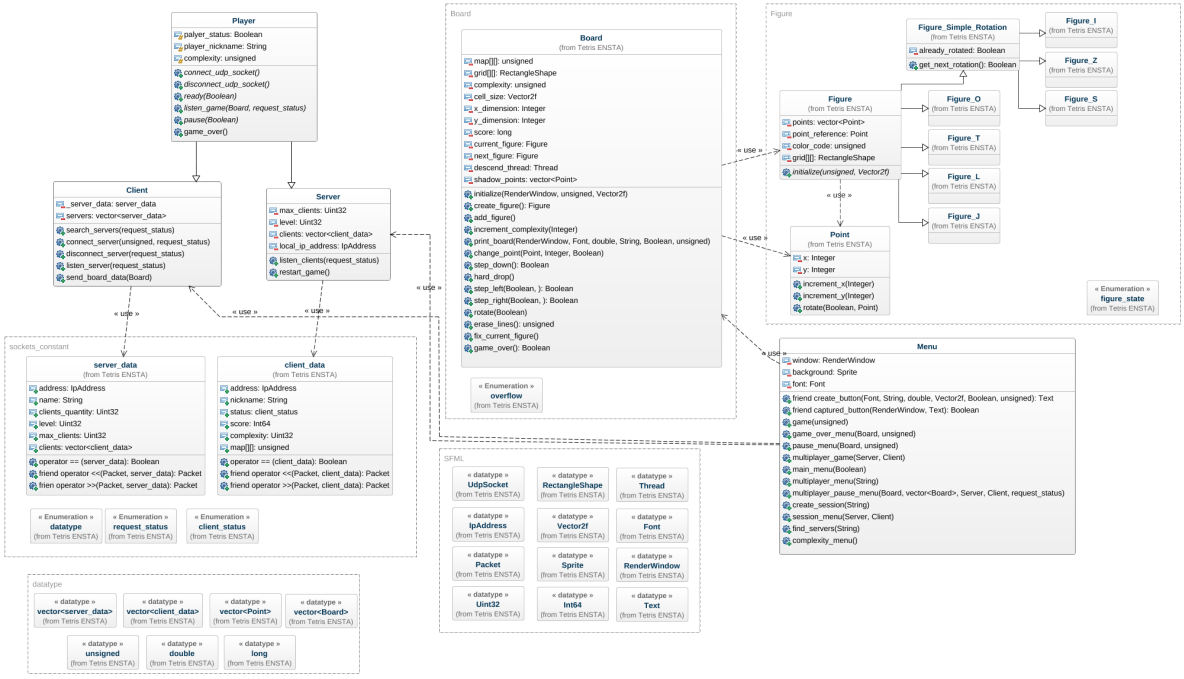


FIGURE 1 – Diagramme de classes

Figure correspond aux menus différents du logiciel et des fonctions qui les lient.

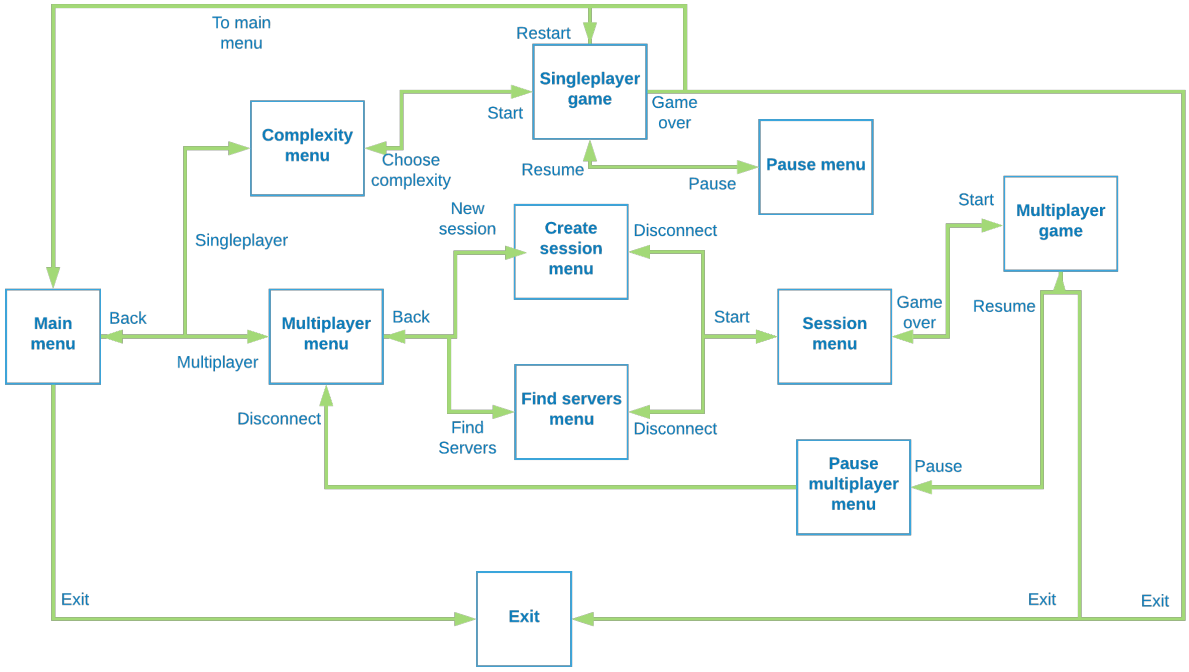


FIGURE 2 – Architecture du menu

# 4 Possibles améliorations

Il existe plusieurs chemins d'évolution du logiciel donné :

1. L'implementation d'un joueur-IA en appliquant des algoritmes d'apprentissage automatique (arbres de décisions, réseaux neurones etc).
2. Création d'un serveur à distance pour pouvoir supporter multiplayer non seulement dans le cadre des réseaux locales et privés.
3. L'integration d'une base des données dans l'application et mis en pratique la fonction de leaderboards / high scores afin que motiver la communauté mondiale.

# Références

Description du projet.