INSTITUT NATIONAL DES SCIENCES APPLIQUÉES ROUEN NORMANDIE

Département STPI

Rapport de projet d'Informatique

Remastered Jeux Classiques

Étudiants

Zouhair	FAKHOURY	zouhair.fakhouy@insa-rouen.fr
Manh Hung	NGUYEN	manh_hung.nguyen@insa-rouen.fr
Tymofii	BILYK	tymofii.bilyk@insa-rouen.fr

15 décembre 2024

Encadré par

Nathalie CHAIGNAUD

Synthèse du projet

Titre du projet : Remastered Jeux Classiques

Groupe : I

Résumé : Le projet présenté dans ce rapport consiste en la création des jeux informatique classiques, développé en langage Pascal. L'objectif principal était de concevoir chaque jeu interactif offrant une expérience utilisateur riche, grâce à des fonctionnalités avancées telles que la personnalisation des niveaux de difficulté, un système de scores, et des thèmes graphiques. L'approche modulaire adoptée dans le développement a permis de structurer le programme de manière claire et évolutive.

Mots clés : Pendu ; TicTacToe ; Puissance 4 ; Pascal

Table des matières

Descriptif du Cahier des Charges Initial

Objectif Principal

Le projet a pour objectif principal la conception et le développement de trois jeux informatiques : **Pendu (Hangman)**,**Tic-Tac-Toe**, et **Puissance 4**. Réalisés en langage Pascal, ces jeux doivent offrir une expérience utilisateur engageante et interactive. L'ensemble repose sur une architecture logicielle permettant une intégration des trois jeux dans leur propre interface indépendante.

Contraintes et Exigences

Le développement repose sur plusieurs contraintes techniques et exigences fonctionnelles qui structurent les différentes étapes du projet :

- **Technologie** : Utilisation du langage de programmation Pascal pour le développement ainsi que ses bibliothèques associées.
- **Architecture du code** : Une organisation modulaire est requise, permettant une évolution simplifiée du projet.
- **Respect des délais** : Trois livraisons progressives doivent être effectuées, chacune respectant des échéances spécifiques.

Fonctionnalités Attendues

Les fonctionnalités prévues dans le cadre du projet incluent :

1. **Pendu**
   - Affichage interactif des lettres déjà devinées, des tentatives restantes, et de la progression graphique du pendu.
   - Interaction via clavier
   - Niveaux de difficulté : facile, moyen, difficile.
   - Statistiques et suivi : tableau de scores et historique des parties.
2. **Tic-Tac-Toe**
   - Grille 3x3 interactive avec affichage dynamique des mouvements.
   - Visualisation en temps réel des positions jouées par chaque joueur.
   - Indicateur de tour actif et détection automatique des victoires ou des égalités.
   - Option de redémarrage rapide sans quitter le jeu.
3. **Puissance 4**
   - Grille 6x7 affichée de manière intuitive
   - Détection automatique des alignements gagnants (horizontaux, verticaux et diagonaux).
   - Fonctionnalité de redémarrage.

Planification des Versions

Pour garantir une évolution fluide du projet, trois versions sont prévues, chacune apportant des ajouts significatifs :

- **Version 1 (18 Octobre) :**
  - Développement du jeu Pendu avec une interface utilisateur simple.
  - Mise en place des niveaux de difficulté : facile, moyen et difficile.
- **Version 2 (16 Novembre) :**
  - Finir le jeu de Pendu
  - Développement des règles, de l'interface, et des fonctionnalités de Tic-Tac-Toe et Puissance 4.
- **Version 3 (9 Decembre) :**
  - Intégration complète du Puissance Tic-tac-toe avec ses règles et son interface. Fin du développement des trois jeux

Livrables

Les éléments à livrer incluent :

- Un code source propre, documenté, et fonctionnel pour les trois jeux.
- Une interface graphique intuitive et prête à l'utilisation.
- Une documentation clair destinée aux utilisateurs finaux.

Conception Globale

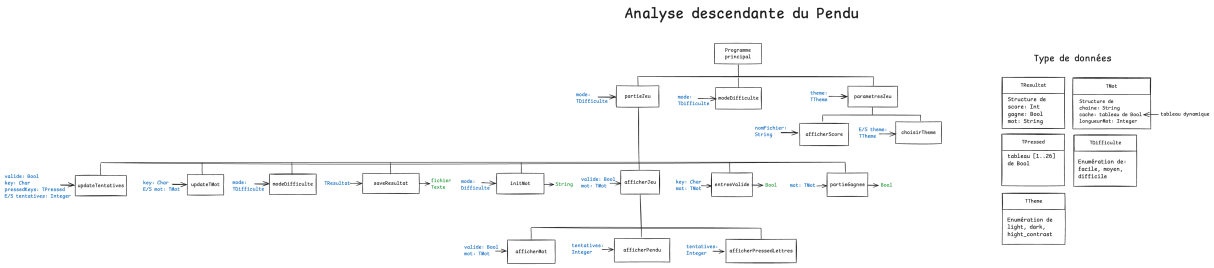

FIGURE 1 – Analyse descendante du Pendu

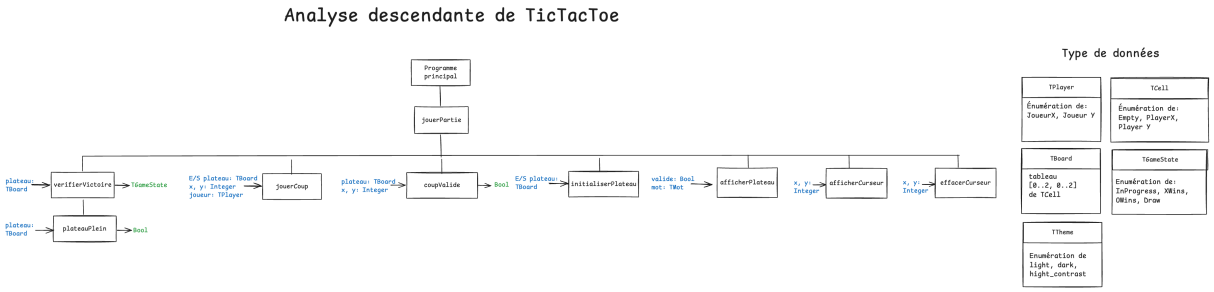

FIGURE 2 – Analyse descendante du TicTacToe

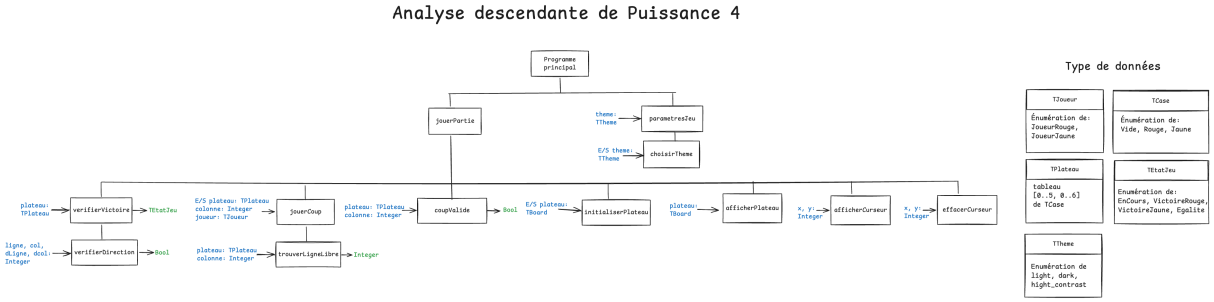

FIGURE 3 – Analyse descendante du Puissance 4

Fonctions et Procédures Clés

Voici une liste des principales fonctions et procédures, accompagnées de leurs signatures et descriptions :

— **Pendu :**

1. `procedure saveResultat(resultat: TResultat)` : Enregistre les résultats d'une partie dans un fichier.
2. `function initMot(mode: TDifficulte): TMot` : Initialise un mot aléatoire à deviner en fonction de la difficulté choisi.

3. `procedure afficherMot(valide: Boolean; mot: TMot)` : Affiche les lettres correctes devinées à l'écran.
4. `procedure afficherPendu(tentatives: Integer)` : Met à jour l'état graphique du pendu selon le nombre d'erreurs.
5. `procedure choisirTheme(var theme: TTheme)` : Permet de sélectionner un thème graphique à appliquer au jeu.

— **Puissance 4 :**

1. `procedure initialiserPlateau(var plateau: TBoard)` : Initialise la grille 6X7 avec des cases vides.
2. `function verifierVictoire(plateau: TPlateau): TEtatJeu` : Vérifie si un joueur a aligné quatre jetons ou si la partie est une égalité.
3. `procedure jouerCoup(var plateau: TPlateau; colonne: Integer; joueur: TJoueur)` : Place un jeton dans la colonne choisie si le coup est valide.
4. `procedure afficherCurseur(position: Integer)` : Montre visuellement la position du curseur sélectionnant une colonne.

— **Tic-tac-toe :**

1. `procedure initialiserPlateau(var plateau: TBoard)` : Initialise la grille 3x3 avec des cases vides.
2. `function verifierVictoire(plateau: TBoard): TGameState` : Vérifie si un joueur a aligné trois symboles ou si la partie est terminée.
3. `procedure jouerCoup(var plateau: TBoard; x, y: Integer; joueur: TPlayer)` : Place un symbole dans une case après avoir vérifié sa validité.
4. `function coupValide(plateau: TBoard; x, y: Integer): Boolean` : Vérifie si une case est vide et dans les limites du plateau.

Type de Données

Le jeu utilise plusieurs types de données pour gérer les informations de manière structurée :

— **Pendu :**

1. `TResultat` : Structure contenant les informations liées à une partie, telles que le mot joué, le score obtenu, et l'état final (gagné ou perdu).
2. `TMot` : Tableau dynamique de caractère représentant le mot à deviner, accompagné d'un tableau booléen pour suivre les lettres déjà découvertes.
3. `TPressed` : Enumération des touches pressées, pendant le jeu par le joueur.
4. `Difficulté` : Enumération des niveaux de difficulté (`facile`, `moyen`, `difficile`).
5. `Thème` : Enumération des thèmes graphiques disponibles (par exemple : `classique`, `sombre`, `coloré`).

— **Puissance 4 :**

1. `TJoueur` :Enumération des deux joueurs avec des couleurs différentes pour distinguer les jetons.
2. `TCase` : Enumération de capacités indique qui occupe une cellule particulière.
3. `TPlateau` : Tableau à deux dimensions, qui correspond à la capacité du plateau. (`7 en longueur` et `6 en hauteur`)
4. `TEtatJeu` : Enumération des états de jeu.
5. `Thème` : Enumération des thèmes graphiques disponibles (par exemple : `classique`, `sombre`, `coloré`).

— **Tic-Tac-Toe :**

1. `TPlayer` :Enumération qui représente les deux joueurs possibles.
2. `TBoard` : Tableau à deux dimensions qui représente le plateau de jeu (`grille 3x3`).
3. `TGameState` : Enumération qui indique l'état actuel de la partie.
4. `TCell` : Enumération qui représente l'état d'une case sur le plateau.

Guide d'Utilisation

Installation

Pour installer et lancer les jeux **Pendu, Puissance 4 ou Tic-Tac-Toe**, suivi les étapes suivantes :

1. **Prérequis logiciels :** Installez un compilateur Pascal (par exemple Free Pascal). What
2. **Installation du programme :** Téléchargez et extrayez l'archive contenant le code source dans un dossier.
3. **Compilation et exécution :** Compilez le programme en ligne de commande ou via un IDE supportant Pascal (Geany). Lancez l'exécutable généré.

Jeu de Pendu

— **But** : Trouvez le mot caché avant d'épuiser vos tentatives.
— **Comment jouer :**
  1. Le mot est masqué par des tirets (_ _ _).
  2. Proposez des lettres avec le clavier : Correct, la lettre affiche. Faux, une tentative est déduite et le pendu avance.
  3. Terminez si toutes les lettres sont trouvées (victoire) ou les tentatives atteignent zéro (défaite).
— **Options supplémentaires :**
  — Difficultés : Facile, moyen, difficile ou chrono.
  — Scores : Consultez-les via le menu.

Tic-tac-toe

— **But** : Alignez trois symboles (X ou O) horizontalement, verticalement ou en diagonale.
— **Comment jouer :**
  1. Une grille 3x3 apparaît.
  2. Les joueurs jouent à tour de rôle. Utilisez les flèches pour bouger le curseur. Placez un symbole avec Entrée.
  3. La partie s'arrête lorsqu'un joueur gagne ou qu'il y a égalité (grille pleine).

Puissance 4

— **But** : Alignez quatre jetons dans une grille 6x7 (horizontal, vertical ou diagonal).

— **Comment jouer :**

1. Une grille 6x7 est visible.
2. Les joueurs jouent à tour de rôle. Utilisez les flèches pour bouger le curseur. Insérez un jeton avec Entrée.
3. Finissez avec un alignement gagnant ou une grille pleine.

Retour sur le Travail de Groupe

Dès le départ, nous avons adopté une approche structurée pour le développement du projet. Après avoir défini les objectifs globaux, nous avons décomposé le projet en blocs fonctionnels distincts, facilitant ainsi la répartition des tâches. Cette méthode a permis de simplifier les étapes complexes en les rendant plus accessibles.

Fortement inspirés par notre expérience précédente en travail de groupe, nous avons rapidement trouvé des méthodes adaptées à nos forces et faiblesses respectives. Nous avons choisi d'utiliser la méthodologie `Scrum`, une méthode agile qui structure le travail en étapes progressives (`Sprints`) de 1 à 2 semaines.

Une communication efficace a joué un rôle central dans la réussite du projet. Nous avons tenu des réunions régulières, à la fois en présentiel et en ligne, pour :
— Planifier les étapes à venir et ajuster les priorités.
— Identifier et résoudre ensemble les problèmes techniques.
— Partager des idées pour améliorer le jeu et son interface.
Les outils principaux utilisés incluent :
— `Discord` pour les réunions virtuelles et les discussions.
— `Google Docs` pour partager des documents collaboratifs.
— `Git` pour suivre les modifications et centraliser le code.
Cette organisation a grandement renforcé la qualité de notre collaboration, favorisant une meilleure intégration des différentes contributions.

Voici la répartition des tâches principales entre les membres de l'équipe :

— **FAKHOURY Zouhair :** Développer principalement TicTocToe à Puissance 4, responsable des fonctionnalités liées au gameplay. Une analyse détaillée des programmes.
— **BILYK Tymofii :** Le développement du pendu, ainsi que et gestion des thèmes visuels. Débogage, tests et établissement de la communication.
— **NGUYEN Manh Hung :** Conception de l'architecture logicielle. Finir et ajouter de fonctionnalités secondaires.

Bien que chaque membre ait eu des responsabilités spécifiques, la flexibilité de notre collaboration nous a permis de travailler ensemble sur les tâches complexes ou imprévues.

Comme tout projet collaboratif, nous avons su surmonter un nombre de défis. L'utilisation du langage Pascal semble aussi très limité par rapport à d'autres langages (manque de bibliothèques, anciennes méthodes non adaptées à l'actualité). Un exemple notable c'est que Pascal ne pouvait pas importer les unités contenues dans des sous-dossiers. Par conséquent, pour des soucis d'organisation et de maintenabilité (ne pas regrouper tous les fichiers des trois jeux dans un même dossier), nous avons décidé de faire tourner nos trois jeux indépendamment. De plus, nous avons décidé que nous n'utiliserons pas de bibliothèques pour afficher les éléments graphiques et gardons le jeu en terminal, car cela ne faisait pas vraiment de différence pour ce type de jeu.

Conclusion

Ce projet a été une expérience particulièrement enrichissante, tant sur le plan technique que personnel. Il nous a permis de mobiliser non seulement nos compétences en développement individuel, mais également nos aptitudes à collaborer efficacement au sein d'une équipe.

Un des aspects les plus marquants a résidé dans la recherche et l'intégration de bibliothèques adaptées, l'optimisation du code, et l'ajout de fonctionnalités particulières. Ces étapes nous a permis de prendre conscience de l'importance d'une architecture logicielle claire et d'un code bien structuré pour assurer la pérennité et la robustesse du projet.

Sur le plan méthodologie, le projet nous a également permis de développer des compétences clés en gestion de projet, comme la répartition de tâches, la gestion des priorités et le respect des échéances. Cependant, l'utilisation du langage Pascal a posé certains défis. Bien qu'elle ait renforcé nos connaissances dans un langage peu utilisé aujourd'hui, cette contrainte a révélé des limites importantes, notamment le manque de bibliothèques adaptées. Pour un projet de cette nature, un langage plus moderne et orienté vers le développement de jeux, tel que **Python** ou **C#**, aurait sans doute été plus approprié.

En conclusion, ce projet a offert une vision précieuse des responsabilités et des défis auxquels sont confrontés les développeurs dans des environnements concrets. Il a consolidé nos compétences techniques tout en renforçant nos aptitudes à travailler en équipe, à prioriser les tâches et à respecter les délais. Ces apprentissages nous seront inestimables pour nos futurs projets et dans notre carrière en développement logiciel.