BORDEAUX INP Ensc

ENSemenC

MAI 2025

Projet	ENSemenC
Date de rendu	23/05/2025

Auteurs
TIEU Mattéo
GERARD Paul

Sommaire

I. Les choix de modélisation des données

Le choix de modélisation des données dans le cadre du projet de simulation de jardin potager a été effectué dans le but de répondre aux besoins spécifiques du jeu et satisfaire les différentes contraintes imposées. Les principaux facteurs décisifs dans ce choix ont été la simplicité, la lisibilité, et la conformité aux principes de la programmation orientée objet appris en cours et en TD. Dans cette première partie, nous explorerons de manière exhaustive les choix faits pour représenter et manipuler les données dans la simulation de jardin.

I.1 Modélisation des plantes et de leur cycle de vie

La représentation des plantes constitue le cœur du système de données de la simulation. Chaque plante est modélisée par une classe contenant les attributs essentiels à son fonctionnement : le type qui identifie l'espèce (tomate, carotte, fraise, etc.), la santé exprimée en pourcentage et initialisée à 100%, les points de croissance qui déterminent le stade de développement, un numéro d'identification unique par type, etc.

```
1   public abstract class Plante
2   {
3       // Compteur pour générer des IDs uniques
4       private static int _compteurId = 0;
5       // Identifiant unique de la plante
6       public int Id { get; private set; }
7       // Nom de la plante
8       public string Nom { get; protected set; }
9       // Santé de la plante (en %)
10      public int Sante { get; set; }
11      // Points de croissance accumulés
12      public int PointsCroissance { get; set; }
13      // Durée de pousse de base en tours
14      public int DureePousseBase { get; protected set; }
15      // Nombre de tours restants avant que la plante atteigne sa maturité
16      public int ToursRestantsAvantMaturite { get; set; }
17      // Type de terrain préféré par la plante
18      public TypeTerrain TypeTerrainAffinite { get; protected set; }
19      // Indique si la plante est malade
20      public bool EstMalade { get; set; }
```

Fig. 1 : Extrait de Plante.cs

Cette approche orientée objet permet une gestion claire et modulaire de chaque plante individuelle. Le système de numérotation automatique facilite l'identification lors des interactions avec le joueur, en attribuant séquentiellement des identifiants comme "Carotte n°1" ou "Tomate n°3". Cette numérotation évite toute confusion lors de la sélection d'actions spécifiques sur une plante particulière.

Le cycle de vie suit quatre états distincts qui correspondent aux phases naturelles d'une plante. L'état de semis représente la phase initiale après plantation, symbolisé par l'icône 🌱. La croissance maintient cette même représentation visuelle jusqu'à l'atteinte de la maturité. L'état de récolte, applicable uniquement aux plantes comestibles, utilise des icônes spécifiques comme 🍅 pour les tomates ou 🥕 pour les carottes. Enfin, l'état de mort survient lorsque la santé atteint 0% ou en cas de non-récolte dans les délais impartis.

```
1   public class Carotte : PlanteComestible
2   {
3       // Constructeur de Carotte
4       public Carotte() : base("Carotte", 4, 6, TypeTerrain.Sableux) { }
5       // Nom, durée de pousse, durée de vie après maturation, affinité terrain
6
7       // Retourne l'icône représentant la carotte
8       public override string ObtenirIcone()
9       {
10          if (EstMature())
11          {
12              return "🥕"; // Icône carotte mature
13          }
14          return "🌱"; // Icône semis/croissance
15      }
16  }
```

Fig. 2 : Exemple de Carotte.cs

Chaque type de plante possède des caractéristiques fixes définies dans une structure de données dédiée. Ces propriétés incluent la durée de croissance nécessaire pour atteindre la maturité, la durée de vie disponible après maturité, la compatibilité saisonnière qui détermine la réaction aux quatre saisons, l'affinité terrain qui spécifie le type de sol optimal, et la vitesse de croissance de base exprimée en points gagnés par tour dans des conditions standard.

I.2 Représentation des terrains et de l'espace de jeu

Le jardin est structuré comme un ensemble fixe de trois terrains distincts, chacun organisé en grille de neuf cases disposées en format 3x3. Cette limitation impose naturellement des choix stratégiques au joueur et maintient un niveau de complexité gérable tout en préservant l'aspect tactique du jeu. Le choix de modéliser les terrains par des tableaux plutôt que des listes a été effectué par simple affinité de l'équipe, et aussi par un sentiment de manque de maîtrise des listes pour les utiliser.

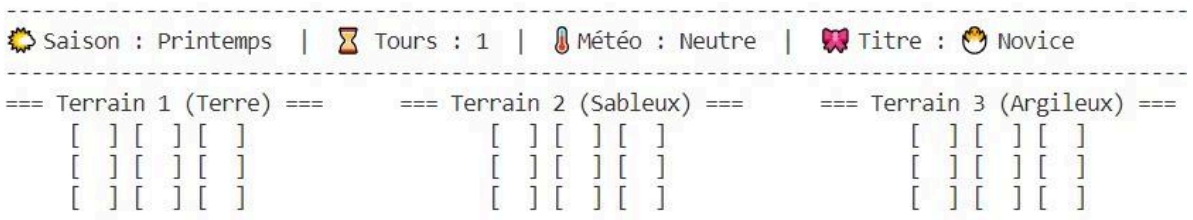

Fig. 3 : Affichage des terrains dans la console

Les trois types de terrains disponibles correspondent à des environnements de culture réalistes. Le terrain sableux favorise certaines plantes comme les carottes, les fleurs ornementales, la lavande et les mimosas. Le terrain argileux convient particulièrement aux pommes de terre et aux tournesols. Le terrain de terre standard représente l'option polyvalente, adaptée aux tomates, pommes, fraises et roses. Cette diversité encourage l'expérimentation et la planification stratégique de l'aménagement du jardin.

La gestion de l'occupation des cases suit un système simple mais efficace. Chaque case peut contenir soit une plante représentée par son icône selon son état, soit de la mauvaise herbe symbolisée par X qui empêche toute plantation, soit demeurer libre et disponible pour une future plantation. Un mécanisme de vérification empêche automatiquement la superposition d'éléments sur une même case, garantissant la cohérence des données.

Le système de compatibilité entre plantes et terrains influence directement les performances de croissance. Un terrain favorable accélère la vitesse de croissance de 50% tout en apportant un bonus de santé de 5% par tour. Un terrain neutre n'apporte aucune modification aux paramètres de base. Un terrain défavorable ralentit la croissance de 50% et

inflige une pénalité de santé de 5% par tour. Cette mécanique encourage la réflexion dans le choix de placement des plantes.

I.3 Gestion du temps et des événements

La modélisation temporelle repose sur un système de tours représentant des semaines, créant un rythme de jeu cohérent avec la réalité agricole. Un cycle complet comprend quatre saisons qui se succèdent automatiquement, chacune modifiant les conditions de croissance selon les affinités spécifiques de chaque plante. Cette approche cyclique apporte une dimension stratégique à long terme au gameplay.

Les événements météorologiques sont représentés par des objets contenant leur type (tempête, canicule, invasion de nuisibles, maladie, mauvaise herbe), leur durée exprimée en nombre de tours d'effet, et leur impact spécifique sur les plantes ou le gameplay. Cette modélisation permet une gestion flexible et extensible des différents événements possibles.

```
1    // Énumération des types d'événements aléatoires
2    public enum TypeEvenement { Aucun, Tempete, Canicule, Nuisibles, Maladie, MauvaiseHerbe }
```

Fig. 4 : Evenements.cs

Chaque événement actif est suivi par un compteur de tours restants qui décrémente automatiquement. Les effets s'appliquent à chaque tour jusqu'à expiration complète de l'événement. Certains événements comme les tempêtes et canicules bloquent temporairement toutes les actions du joueur, simulant l'impossibilité d'intervenir durant des conditions météorologiques extrêmes.

Le passage en mode urgence constitue une mécanique particulière déclenchée par des événements spécifiques, principalement les invasions de nuisibles. Ce mode modifie temporairement l'interface utilisateur et restreint les actions disponibles à des choix d'urgence spécialisés. Une fois l'événement résolu, le système restaure automatiquement le mode classique, permettant la reprise normale du gameplay.

I.4 Système de score et de progression

Le calcul du score suit une formule simple et transparente qui récompense les réussites tout en pénalisant les échecs. Chaque plante récoltée rapporte 10 points tandis que chaque plante morte coûte 5 points. Le score final correspond donc au calcul suivant : (nombre de récoltes × 10) - (nombre de morts × 5). Cette approche encourage la réussite tout en maintenant une marge d'erreur acceptable pour les débutants.

Le système de titres structure la progression du joueur autour de trois paliers distincts. Le titre de "🐣 Novice" est attribué par défaut à tous les joueurs au début de la partie. Le titre de "🌱 Jeune pousse" est débloqué dès l'atteinte de 50 points cumulés. Enfin, le titre de "🏅 Maître du potager" récompense les joueurs ayant atteint ou dépassé 100 points. Cette progression graduelle maintient la motivation sur le long terme.

```
12        // Constructeur de TitreJoueur
13        public TitreJoueur()
14        {
15            NomTitre = "🐣 Novice"; // Titre au début du jeu
16            scoreAtteintPourTitre = 0;
17            Affichage = new Affichage();
18        }
19
20        // Met à jour le titre du joueur en fonction du score
21        // Un titre obtenu ne peut pas être perdu même si le joueur ...
22        public void MettreAJourTitre(int scoreActuel)
23        {
24            if (scoreActuel >= 100 && scoreAtteintPourTitre < 100)
25            {
26                NomTitre = "🏅 Maître du potager";
27                scoreAtteintPourTitre = 100;
28            }
29            else if (scoreActuel >= 50 && scoreAtteintPourTitre < 50)
30            {
31                NomTitre = "🌱 Jeune pousse";
32                scoreAtteintPourTitre = 50;
33            }
34        }
```

Fig. 5 : Extrait de TitreJoueur.cs

La gestion de la progression s'effectue en temps réel durant toute la session de jeu. L'attribution d'un nouveau titre déclenche automatiquement l'affichage d'un message de félicitations unique, évitant les répétitions lors des tours suivants. Le titre actuel du joueur

reste visible en permanence dans l'interface principale, renforçant le sentiment d'accomplissement.

I.5 Interactions entre les éléments du système

L'impact des saisons sur les plantes constitue l'un des mécanismes centraux du système. Chaque plante réagit différemment selon la saison courante, créant des fenêtres optimales pour certaines cultures. Une saison favorable accélère la vitesse de croissance de 50% tout en apportant un bonus de santé de 5% par tour. Une saison neutre n'entraîne aucune modification des paramètres de base. Une saison défavorable ralentit la croissance de 50% et impose une pénalité de santé de 5% par tour.

Les événements météorologiques modifient temporairement l'ensemble des conditions de culture. Une météo avantageuse apporte un bonus de santé de 10% par tour à toutes les plantes du jardin. À l'inverse, une météo désavantageuse inflige une pénalité de 10% par tour. Les événements bloquants comme les tempêtes empêchent complètement toute action du joueur, simulant l'impossibilité d'intervenir dans de telles conditions.

L'influence des terrains s'applique en permanence et se cumule avec tous les autres modificateurs environnementaux. Cette interaction continue entre l'affinité terrain/plante, les conditions saisonnières et les événements météorologiques crée un système complexe nécessitant une planification stratégique approfondie. Le joueur doit considérer simultanément tous ces facteurs pour optimiser ses choix de plantation et de gestion.

Les mécanismes de santé et de croissance fonctionnent en parallèle selon des règles distinctes mais complémentaires. Le système de croissance repose sur l'accumulation progressive de points jusqu'à l'atteinte du seuil de maturité, ce processus étant modifié par l'ensemble des facteurs environnementaux. Le système de santé évolue continuellement selon les conditions actuelles et détermine la survie de la plante. Les modificateurs de vitesse s'appliquent de manière multiplicative tandis que les effets sur la santé s'additionnent tour après tour, créant une dynamique riche et réaliste.

II. Structuration du code

II.1 Description des sous-programmes principaux

Le cœur de l'application est organisé autour de plusieurs modules fonctionnels, chacun encapsulant une logique métier distincte.

II.1.a Gestion de l'affichage et de l'interface utilisateur

Cette catégorie de sous-programmes est essentielle pour l'interaction avec le joueur. Elle comprend les fonctions dédiées à la visualisation du jardin, qu'il s'agisse de la vue synthétique offrant un aperçu global ou de l'affichage détaillé des plantes avec leurs attributs spécifiques. L'interface de navigation et les menus interactifs permettent au joueur de choisir ses actions, tandis que la gestion des messages d'information et d'alerte assure une communication claire et immédiate sur l'état du jeu ou les événements survenus. Enfin, l'affichage du score et des statistiques fournit au joueur un retour sur sa progression et ses performances.

II.1.b Initialisation et configuration du jeu

Au démarrage de la simulation, des sous-programmes spécifiques se chargent de préparer l'environnement de jeu. Ils sont responsables de la création du jardin lui-même, y compris la mise en place des différents types de terrains. L'initialisation des variables de jeu comme la saison actuelle, les conditions météorologiques ou le score initial est également gérée ici. De plus, ces routines prennent en charge la configuration des paramètres de chaque type de plante, définissant leurs caractéristiques de croissance, de santé et leurs compatibilités.

II.1.c Gestion des actions du joueur

Ces sous-programmes orchestrent toutes les interactions directes du jardinier virtuel avec son potager. Cela inclut la plantation de semis dans les cases disponibles, l'arrosage et l'entretien des plantes pour maintenir leur santé, le désherbage des cases envahies, et le traitement des maladies affectant les cultures. Lorsque les plantes arrivent à maturité, des fonctions dédiées permettent la récolte des plantes matures. En revanche, bien qu’une partie du code destiné à être utilisé à cet effet soit présent dans le dépôt GitHub final, le

joueur ne peut pas installer des infrastructures telles que des serres ou des barrières, car le code n’a pas pu être rendu fonctionnel..

II.1.d Système de simulation temporelle

La progression du temps est un pilier du jeu, et ces sous-programmes en sont les garants. Ils gèrent le déroulement des tours et le changement de saisons, impactant directement l'environnement du jardin. Des calculs précis sont effectués pour la croissance des plantes à chaque tour, prenant en compte les divers modificateurs. Parallèlement, la gestion de la santé des plantes est assurée, actualisant leur vitalité en fonction des conditions et des actions du joueur.

II.1.e Gestion des événements et du mode urgence

Pour dynamiser le jeu, des sous programmes spécifiques gèrent les imprévus. Ils s'occupent de la génération aléatoire d'événements tels que les tempêtes ou les invasions de nuisibles. Lorsqu'un événement critique survient, une transition vers le mode urgence est activée, modifiant temporairement les mécaniques de jeu. Dans ce mode, le joueur se voit proposer des actions spécifiques d'urgence, et les sous-programmes calculent ensuite les conséquences de ses choix ou de son inaction sur le jardin.

II.1.f Validation et contrôle de jeu

Une série de sous programmes veille à la cohérence et à la validité des opérations. Ils effectuent la vérification des conditions de plantation, s'assurant qu'une plante peut être placée sur un terrain donné. La validation des actions possibles garantit que le joueur ne tente pas une action inapplicable à un instant T. Ces routines contrôlent aussi l'état des plantes, notamment pour détecter la mort ou la maturité. Pour finir, la gestion des limites, qu'il s'agisse du nombre d'actions par tour ou des cases disponibles, est prise en charge pour maintenir l'équilibre du jeu.

II.2 Dynamique d'exécution

La dynamique d'exécution du simulateur de jardin est conçue pour offrir une expérience de jeu cohérente et réactive, orchestrant la succession des événements et les interactions du joueur. L'ensemble du programme s'articule autour d'une boucle principale qui gère la progression du temps et l'alternance entre les différents modes de jeu.

II.2.a Flux principal du programme

Le cœur du jeu repose sur une boucle de jeu principale qui représente les tours successifs de la simulation. Chaque itération de cette boucle correspond à l'écoulement d'une période de temps, généralement une semaine dans le mode classique, pendant laquelle les plantes évoluent et des événements peuvent survenir.

La séquence d'un tour standard est bien définie. Elle débute par la mise à jour des paramètres environnementaux (saison, météo), suivie par la croissance et l'évolution de la santé de toutes les plantes. Ensuite, le système vérifie l'apparition éventuelle de nouveaux événements ou mauvaises herbes. C'est à ce moment que le joueur peut interagir avec son jardin en réalisant ses actions. Une fois les actions effectuées ou le quota d'actions épuisé, le score est mis à jour et le tour se termine, préparant le terrain pour le suivant.

La gestion des interruptions est un aspect crucial de cette dynamique. Le mode urgence, par exemple, interrompt temporairement le flux standard pour que le joueur puisse réagir à des situations critiques.

II.2.b Gestion des modes de jeu

Le jeu alterne entre deux modes de jeu principaux, chacun avec ses propres spécificités. Le mode classique constitue le cadre de jeu par défaut. Il permet au joueur de gérer son potager à un rythme normal, en planifiant ses cultures et en réalisant les actions d'entretien courantes. Les mécaniques de base du jeu, comme la croissance régulière des plantes et la gestion des ressources, s'y déroulent naturellement.

Par contraste, le mode urgence est déclenché par des événements spéciaux qui requièrent une attention immédiate du joueur. Dans ce mode, le temps est simulé de manière plus condensée, les actions sont rapides et les conséquences sont souvent plus directes. Les

transitions entre les modes sont fluides : un événement aléatoire peut basculer le jeu en mode urgence, et une fois la crise résolue, la simulation reprend son cours en mode classique.

II.2.c États du jeu et conditions de fin

La dynamique du jeu est également définie par le suivi de l'état global du jardin. Cela inclut la surveillance de la santé de chaque plante, la disponibilité des cases de terrain, et l'impact des événements en cours. Les conditions de progression, comme l'atteinte de certains seuils de score ou l'obtention de titres honorifiques, sont constamment évaluées, offrant au joueur des objectifs à long terme. Bien que les spécifications n'imposent pas une fin de partie stricte, le système est conçu pour pouvoir intégrer une gestion de la fin de partie si des objectifs ultérieurs venaient à être ajoutés.

II.2.d Interactions utilisateur et réactivité

L'interactivité est au cœur de la dynamique. Le traitement des choix du joueur est immédiat : chaque sélection dans les menus est prise en compte sans délai. La validation et l'exécution des actions sont réalisées en temps réel, avec des vérifications automatiques pour s'assurer que les actions sont valides dans le contexte actuel du jeu. Les retours d'information en temps réel sont constants, qu'il s'agisse de messages de confirmation après une action, d'alertes en cas d'événement, ou de l'affichage mis à jour du jardin, garantissant que le joueur est toujours informé de l'état de son potager.

III. Organisation de l'équipe, répartition des tâches et planning

III.1 Organisation de l'équipe

Après une première session de travail en binôme, l'équipe a décidé de s'orienter vers une collaboration principalement à distance pour la suite du projet de simulation de jardin. Nous pensions pouvoir capitaliser sur les outils de gestion de version comme GitHub, où chaque avancée serait partagée via des branches annexes, permettant une synchronisation efficace de nos efforts. La communication a été maintenue quotidiennement par messagerie, complétée par une à deux réunions hebdomadaires. Ces échanges étaient cruciaux pour partager les idées de fonctionnalités, discuter des manières originales d'implémenter certaines parties du jeu, et s'accorder sur les orientations techniques avant de répartir les tâches pour la semaine à venir.

Malheureusement, après une première phase d'utilisation de GitHub, le codage partagé en ligne a un peu été laissé de côté. Il est donc vrai que le codage intensif n'a réellement démarré que deux semaines avant la date limite. Mais cette temporalité peut se justifier en partie par la nature même du projet, qui exigeait une fondation solide et une compréhension approfondie des principes de la programmation orientée objet. Face à cette complexité, nous avons décidé de prendre le temps nécessaire pour établir et affiner entièrement les spécifications de notre simulation de jardin avant de nous plonger dans l'écriture du code. Cette phase de conception préliminaire, bien que consommatrice de temps, s'est avérée indispensable pour poser des bases saines et éviter des problèmes majeurs par la suite. Cependant, il est vrai que l'intensité des deux dernières semaines intensives au niveau du projet, agrémentées par une période de rendus de projets et de révision pour les partiels, ne nous ont pas permis d'utiliser GitHub comme on avait prévu de le faire. Voyant le délai de la fin du projet approcher, nous avons préféré travailler ensemble la plupart du temps, notamment grâce à l'extension LiveShare de VSCode, jusqu'à obtenir un code satisfaisant pour le projet. Nous n'avons ainsi push sur le dépôt GitHub que les codes fonctionnels ou quasi-fonctionnels, au fur et à mesure de la dernière semaine.

III.2 Répartition des tâches dans le groupe

Nous avons tous les deux réfléchi à la modélisation des plantes et de leur cycle de vie, définissant les structures de données pour leurs attributs (santé, croissance, maturité) et les règles de leur évolution. La représentation des terrains et de l'espace de jeu, incluant la structure du jardin et le système de compatibilité plantes-terrains, relevait également de ses attributions. Cette personne a aussi conçu le système de scoring et de progression, intégrant le calcul des points et la logique d'attribution des titres honorifiques. Enfin, elle a supervisé la modélisation des interactions complexes entre les éléments du système, comme l'impact croisé des saisons, de la météo et des terrains sur les plantes, et a défini les mécanismes fondamentaux de santé et de croissance. Pour la structuration du code, elle était la garante de la conception du flux principal du programme et des conditions de fin de jeu, ainsi que de la validation des actions possibles au niveau de la logique métier.

Ensuite nous nous sommes divisés les tâches, une première personne s'est concentrée sur tout ce qui touche à la gestion des messages d'information et d'alerte, ainsi que l'affichage du score et des statistiques, étaient également sous sa responsabilité. Concernant les actions du joueur, cette personne a implémenté le traitement des choix du joueur et la validation des actions au niveau de l'entrée utilisateur, garantissant des retours d'information en temps réel. La seconde personne eut pour mission la présentation du jeu et à sa réactivité, mais aussi à la temporalité et aux événements. Elle était en charge de la gestion de l'affichage et de l'interface utilisateur, ce qui inclut l'affichage synthétique et détaillé du jardin, la construction des menus interactifs et de l'interface de navigation. Également de gérer les imprévus dans le jardin, développant la gestion du temps et des événements, modélisant le système de tours et de saisons, et impliquant la représentation ainsi que les effets des événements météorologiques. La gestion de la durée et des conséquences des événements, ainsi que la transition entre les modes classique et urgence, figuraient parmi ses principales responsabilités. Elle a été le moteur du système de simulation temporelle, calculant la progression des tours, la croissance et la santé des plantes à chaque itération.

III.3 Planning de réalisation

Le planning de réalisation a été structuré pour permettre une progression du projet, avec des objectifs clairs à chaque étape.

Le projet a été découpé en trois phases principales :

1. Phase d'initialisation et modélisation : Définition des classes de base (Plante, Terrain), mise en place de la structure du jardin, et affichage initial rudimentaire.
2. Phase de Développement des Mécaniques Centrales : Implémentation de la croissance, de la santé des plantes, des actions du joueur (planter, arroser, récolter), et du système de tours.
3. Phase d'amélioration et d'intégration: Ajout des événements météorologiques, du mode urgence, du scoring, des titres, des infrastructures et du raffinement de l'interface utilisateur.

Nous avions défini des jalons et livrables intermédiaires à la fin de chaque phase, nous permettant de valider l'avancement et d'apporter les corrections nécessaires. Par exemple, à la fin de la phase 1, un jardin statique avec des terrains distincts était fonctionnel. À la fin de la phase 2, le cycle de vie des plantes et les actions de base du joueur étaient opérationnels.

IV. Bilan

Le développement de cette simulation de jardin a été une expérience riche en apprentissages, bien qu'éprouvante par sa difficulté. Ce bilan permet de dresser un tableau des difficultés rencontrées, des réussites et des pistes d'amélioration pour l'avenir du projet.

IV.1 Les défis durant le développement

Plusieurs aspects du projet ont présenté des défis significatifs, nécessitant une réflexion approfondie et des solutions adaptées. La complexité de la modélisation des interactions multiples a été l'un des premiers obstacles. Intégrer l'influence du terrain, de la saison et de la météo sur la croissance et la santé de chaque plante a demandé une conception minutieuse des attributs et des méthodes. Il a fallu s'assurer que chaque facteur interagit de manière logique et prévisible, sans créer de comportements aberrants ou imprévus.

La gestion de la cohérence des données dans le temps a également représenté un enjeu. Avec des tours successifs et des événements aléatoires, maintenir un état précis et à jour pour chaque plante (santé, croissance, statut) et pour l'environnement (saison, météo actuelle) exige une attention constante à la synchronisation des données et à la robustesse des mécanismes de mise à jour.

L'équilibrage du gameplay s'est avéré être un processus itératif. Déterminer la fréquence appropriée des événements météorologiques ou des invasions de nuisibles, ainsi que les vitesses de croissance idéales pour chaque plante, a nécessité de nombreux tests. L'objectif était de rendre le jeu suffisamment stimulant sans être frustrant, et de garantir que la progression du joueur soit gratifiante.

Également, concevoir une interface utilisateur intuitive pour un jeu en console a été un défi particulier. Sans l'avantage des éléments graphiques, il a fallu s'appuyer uniquement sur le texte et les emojis pour représenter le jardin, les plantes et les menus. L'organisation des informations, la clarté des messages et la simplicité de la navigation étaient primordiales pour offrir une expérience utilisateur agréable malgré les limitations techniques de la console.

IV.2 La plus grande surprise lors du développement

Au cours du développement, la plus grande surprise a résidé dans la complexité sous-estimée de la gestion des événements aléatoires et de leurs impacts en mode urgence. Initialement, l'idée semblait simple : déclencher un événement et appliquer des conséquences. Cependant, s'assurer que ces événements s'intègrent harmonieusement dans le cycle de jeu, qu'ils ne bloquent pas le joueur de manière injuste, et que leurs effets soient réversibles ou gérables, a demandé une ingénierie plus poussée que prévu. La transition fluide entre le mode classique et le mode urgence, ainsi que la gestion des actions spécifiques à l'urgence et de leurs répercussions (comme l'épuisement du joueur), ont exigé une logique conditionnelle détaillée et une attention particulière aux états du jeu. Cette complexité a mis en lumière l'importance d'anticiper les interactions entre les différentes mécaniques de jeu.

Au-delà de la gestion des évènements aléatoires, le système d’infrastructures nous a posé plus de problèmes que prévu. En effet, bien que présentes dans notre code et sur l’affichage dans la console lors du déroulement du jeu, les infrastructures ne sont pas fonctionnelles.

IV.3 Résultats et perspectives

Malgré les défis, le projet a atteint la plupart de ses objectifs initiaux.

En termes de fonctionnalités implémentées vs prévues, nous avons réussi à intégrer l'ensemble des mécaniques clés : le cycle de vie des plantes avec leurs caractéristiques spécifiques, les différents types de terrains et leurs affinités, le système météorologique et saisonnier, les actions du joueur (planter, arroser, récolter, désherber, traiter), les outils et le système de scoring avec les titres. Le mode urgence et la gestion des événements imprévus ont également été implémentés avec succès, ajoutant une couche de dynamisme au jeu.

La qualité du code et le respect des principes de la POO ont été des priorités constantes. L'encapsulation a été appliquée pour protéger l'état interne des objets, l'héritage a permis de modéliser les différents types de plantes et de terrains à partir de classes de base, et le polymorphisme a été utilisé pour gérer les comportements variés des plantes et des

événements de manière unifiée. Le code est structuré en modules logiques, favorisant la lisibilité et la maintenabilité.

Pour une version future, plusieurs améliorations sont envisageables. D’abord il faudrait rendre la gestion des infrastructures fonctionnelle, cela apporterait un meilleure équilibre au jeu et permettrait une meilleure expérience pour l’utilisateur. L'ajout de nouvelles variétés de plantes avec des caractéristiques uniques, l'introduction d'un système économique pour acheter des semis ou des outils, ou encore la possibilité d'agrandir le jardin, pourraient enrichir l'expérience de jeu. Une interface graphique, même simple, pourrait également améliorer grandement l'immersion et la fluidité du jeu.