МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Чорноморський національний університет імені Петра Могили
Факультет комп'ютерних наук
Кафедра інженерії програмного забезпечення

Чимбір Сергій Миколайович

УДК 004.51

МОБІЛЬНА ГРА-СИМУЛЯТОР ЕКОСИСТЕМИ ДИКОЇ ПРИРОДИ З ВИКОРИСТАННЯМ ШТУЧНОГО ІНТЕЛЕКТУ

Автореферат кваліфікаційної роботи на здобуття
ступеня вищої освіти «Бакалавр»

Спеціальність 121 «Інженерія програмного забезпечення»

Освітня кваліфікація
«Бакалавр з інженерії програмного забезпечення»

Миколаїв – 2021

Кваліфікаційною роботою є рукопис.

Робота виконана в Чорноморському національному університеті імені Петра Могили Міністерства освіти і науки України на кафедрі інженерії програмного забезпечення.

Науковий керівник: канд. техн. наук доцент (б.в.з.) кафедри інженерії програмного забезпечення Горбань Гліб Валентинович.

Рецензент: кандидат фізико-математичних наук доцент кафедри інтелектуальних інформаційних систем Кулаковська Інесса Василівна.

Захист відбудеться 24 червня 2021 р. о 9 год. на засіданні екзаменаційної комісії (ауд. 2 309) у Чорноморському національному університеті імені Петра Могили за адресою: вул. 68 Десантників, 10, м. Миколаїв, 54003.

З кваліфікаційною роботою можна ознайомитися в бібліотеці Чорноморського національного університету імені Петра Могили за адресою: вул. 68 Десантників, 10, м. Миколаїв, 54003.

Автореферат представлений «18» червня 2021 р.

Секретар
екзаменаційної комісії,
старший викладач І. О. Кандиба

ЗАГАЛЬНА ХАРАКТЕРИСТИКА РОБОТИ

Актуальність роботи визначається необхідністю спонукати дітей вчитись та розвиватись за допомогою сучасних технологій. У зв'язку зі стрімким розвитком технологій, рано чи пізно у всіх навчальних закладах будуть використовувати підхід, який завдяки іграм та за стосункам, значно покращить рівень навчання дітей.

Об'єктом роботи є процеси організації ігрової діяльності дітей молодших класів із використанням сучасних технологій.

Предметом роботи є методи штучного інтелекту що забезпечують імітацію поведінки тварин у природі.

Метою кваліфікаційної роботи є роботи є створення імітації штучного інтелекту тварин для наочної демонстрації процесів в екосистемі дикої природи.

Практичним значенням отриманих результатів роботи полягає у можливості застосування отриманої гри у навчальних закладах, яка зробить навчання дітей різноманітнішим та цікавішим. Це допоможе дітям початкових класів краще засвоїти тему екосистеми у дикій природі.

КРБ містить 3 розділи та спеціальну частину, 49 ілюстрацій, 8 таблиць, 17 джерел в переліку посилань.

ОСНОВНИЙ ЗМІСТ РОБОТИ

У **вступі** подано загальну характеристику досліджуваної теми, обґрунтовано актуальність дипломної роботи, сформульовано мету та завдання досліджень, вказано методи та засоби дослідження.

У **першому** розділі був аналіз предметної сфери, у якому описувався штучний інтелект, а саме той, який використовується для використання у відеоіграх.

В нашому випадку штучний інтелект призначений для моделювання поведінки тварин. У всіх комп'ютерних іграх використовується штучний інтелект для відображення поведінки різних персонажів, тварин, роботів та інше. Чим якісніше буде прописаний штучний інтелект, тим правдоподібніше буде виглядати ігровий світ. На це виділяють багато коштів, сил та часу. Це дуже важка робота. Треба проводити досліди, спостерігати за повадками тварин. Також важливим є порядок пріоритетів. На приклад, для тварини буде пріорітетніішим зайти воду, аніж знайти собі їжу. Також, на приклад, хижа тварина повинна розуміти, у яких випадках треба нападати на тварин або людей, чи зможе він подолати свою здобич та чи не стане він сам здобиччю.

У кожної тварини є шаблон поведінки, який включає набір станів тваринни і правила переходу між ними. У різних тварин може бути різний набір станів, який визначає, як ця тварина буде себе вести в різних ситуаціях.

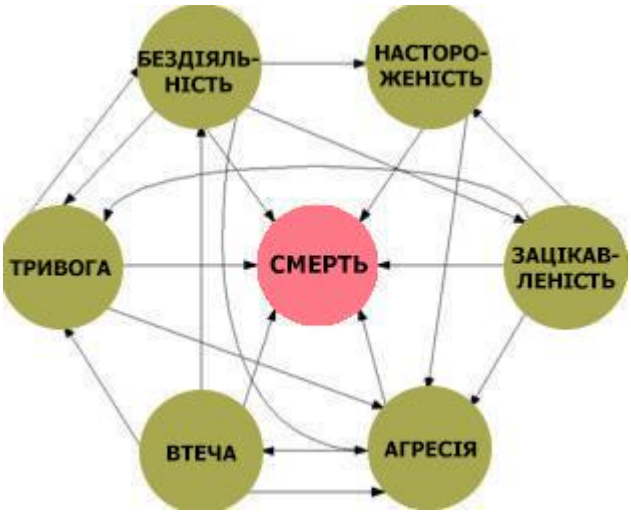

Рисунок 1 – Схема станів тварин та переходів між ними

Також був огляд та аналіз наявних публікацій. Було знайдено чотири гри, які були схожі за темою та реалізацію. Кожна гра була індивідуальною. Вони мали свої як переваги так і недоліки. Також у цих іграх було багато функціоналу. Були наведені такі ігри як: Among Ripples: Shallow Waters, SimLife, Orb Farm та SimEarth.

Не мало важливим пунктом було зрозуміти що із себе представляє процедурна генерація світу, як краще його робити та де використовується вона.

Щоб гра симулятор екосистеми не була одноманітною, було прийнято рішення створити генератор ігрової зони. Це складний етап розробки через те, що в нашому випадку потрібно буде самому створювати алгоритми та умови, за якими буде генеруватися ігрова зона.

Дуже часто можна почути, що відеоігри це марна трата часу, не потрібне заняття, аніж пограти у відеогру – краще піти почитати книгу. Але, щоб там не казали, практика доводить, що ігри сприяють розвиненню тих здібностей, які не отримаєш якщо вчити предмети тільки за книжками. Саме тому був виділений підрозділ під тему впливу відеоігор на розвиток дітей.

У даному розділі також були описані специфікації вимог до програмного забезпечення, де були розглянуті основні функції та вимоги до програмного забезпечення.

Щоб застосунок був правильним, гарним, зрозумілим та якісним, він повинен реалізовувати головний архітектурний паттерн MVC(Model-View-Controller) – архітектурний шаблон, який зазвичай використовується для розробки користувацьких інтерфейсів, який ділить додаток на три взаємопов'язані частини. Це робиться для того, щоб відокремити внутрішні подання інформації від способів, якими інформація представляється і приймається від користувача.

Шаблон проєктування MVC роз'єднує ці основні компоненти, повторно використовувати код і паралельну розробку. Традиційно використовується для настільних графічних інтерфейсів користувача (GUI), ця архітектура стала популярною для проєктування веб-додатків. Популярні мови програмування,

такі як JavaScript, Python, Ruby, PHP, Java і C#, мають фреймворки MVC, які використовуються в розробці веб-додатків.

Модель має такі три частини:

- модель (дані, що відокремлені від інтерфейсу користувача)
- контролер (дозволяє впливати на вигляд та передавати дані для обробки моделі, що були створені діями користувача)
- вигляд (графічний інтерфейс користувача)

Цей розділ був закінчений висновком, який коротко описав все, що було розглянуто в даному розділі.

Другий розділ був присвячений проектуванню. У ньому в основному розроблялись діаграми, які в свою чергу допомогли спроектувати застосунок.

Спочатку були створені usecase. А саме три види: коротка, поверхнева та повна.

1. Коротка – короткий опис в один абзац одного зі сценаріїв (зазвичай успішного) роботи системи. Виконуються під час початкового аналізу вимог до системи;
2. Поверхнева – поверхневий опис у вільній формі усіх сценаріїв (головного і альтернативних) одного з usecases. Виконуються під час початкового аналізу вимог до системи.
3. Повна – всі кроки і дії детально описані, включаючи перед та пост умови виконання usecase Виконуються на стадії відбору з повного переліку usecases в короткій та поверхневій формах невеликої частини важливих (критичних для роботи системи) usecases.

У наступному підрозділі були розглянуті діаграми взаємодії. Діаграма взаємодії – це діаграма, на якій представлено взаємодію, що складається з безлічі об'єктів і відносин між ними, включаючи повідомлення, якими вони обмінюються. Ось наведена одна з діаграм:

Діаграма взаємодії «Генерація локації»

При (1)натисканні на кнопку «Генерація», (2)завантажується вікно меню генерації. Система (3)задає значення за замовчуванням, система їх (4)зберігає. Користувач (5)обирає інший стиль локації, система (6)перезаписує значення

поля, та (7)зберігає його. Користувач (8)обирає розмірність локації, система (9)перезаписує значення поля, та (10)зберігає його. Користувач (11)обирає кількість рослин, система (12)перезаписує значення поля, та (13)зберігає його. Користувач (14)обирає кількість хижаків, система (15)перезаписує значення поля, та (16)зберігає його. Користувач (17)обирає кількість травоїдних, система (18)перезаписує значення поля, та (19)зберігає його. Користувач натискає кнопку (20) «Згенерувати». Система за алгоритмами (21)генерує локацію.

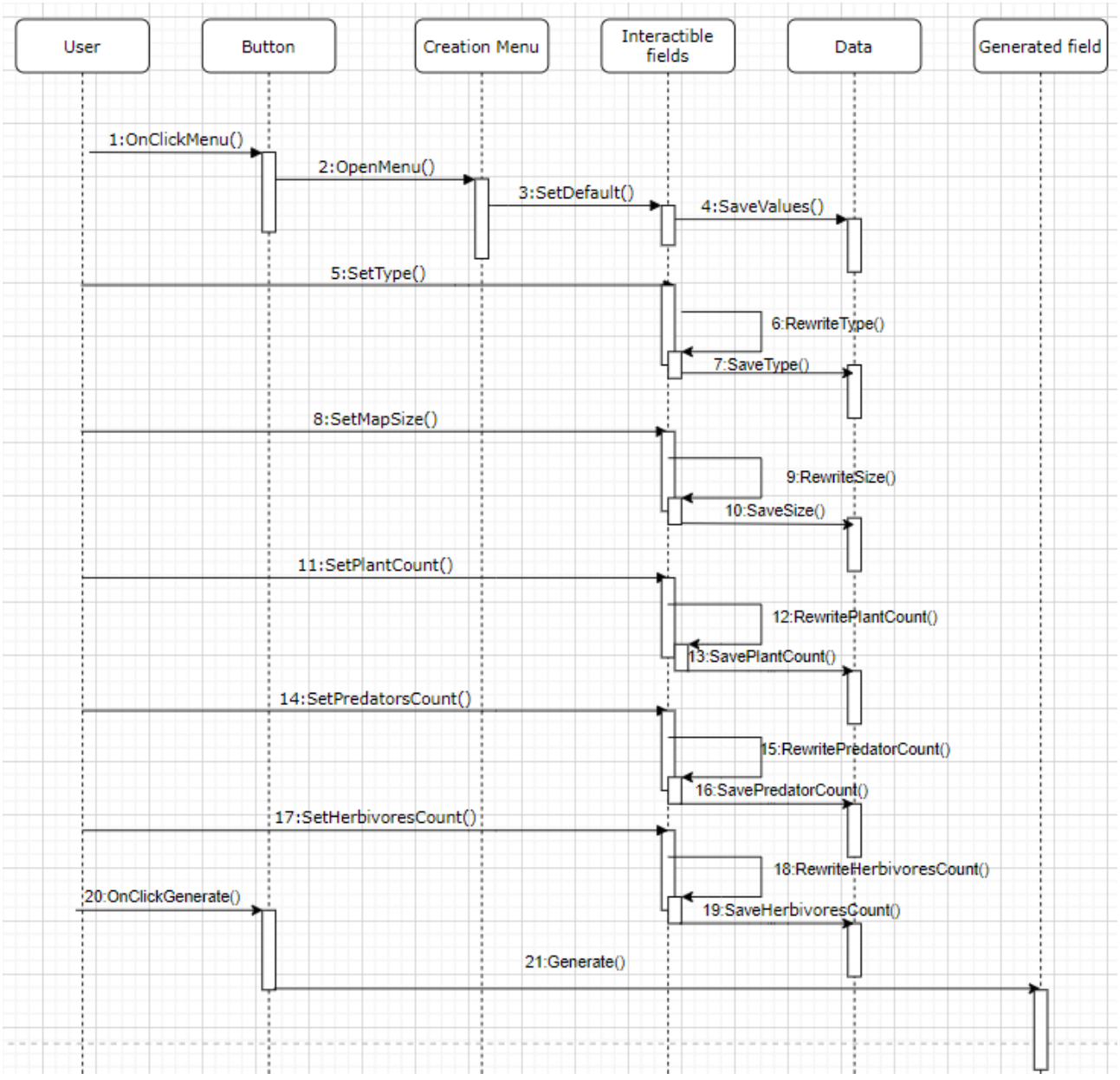

Рисунок 2 – Діаграма взаємодії «Генерація локації»

Наступним підпунктом цього розділу були розглянуті діаграми варіантів використання. Діаграма варіантів використання - це вихідне концептуальне уявлення або концептуальна модель системи в процесі її проєктування і

розробки. В проєкті було створено чотири діаграми варіантів використання. Один із яких є діаграма варіантів використання для додавання генерації зони.

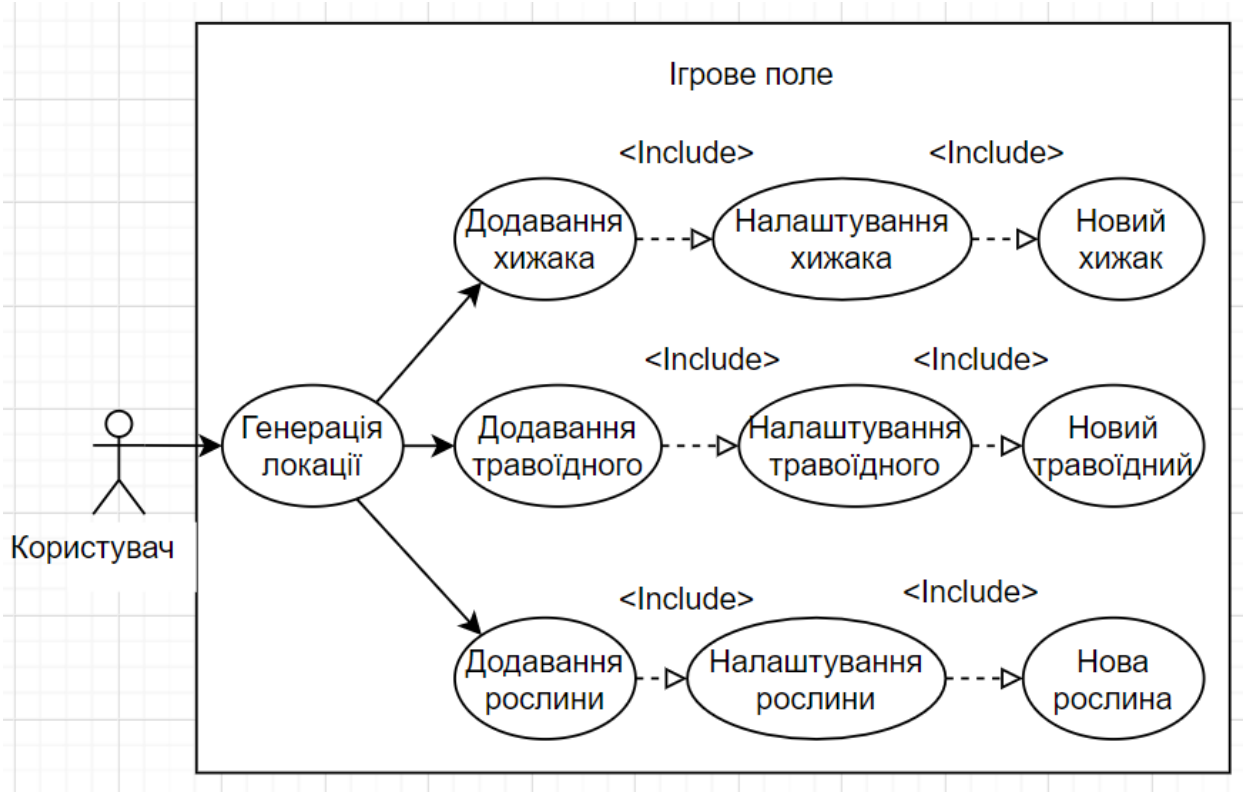

Рисунок 3 – Діаграма варіантів використання для додавання тварин та рослин

Наступним пунктом були розглянуті алгоритми використання. В основному в проєкті є декілька алгоритмів використанн. У нашому випадку таких діаграм п’ять. Вони описують налаштування генератору ігрової зони, додавання хижака, травоїдних та рослин. Завжди є початок та кінець діаграми, на діаграмі є різні види геометричних фігур, кожен з яких має свою функцію. Далі приклад усіх алгоритмів використання, що є у проєкті.

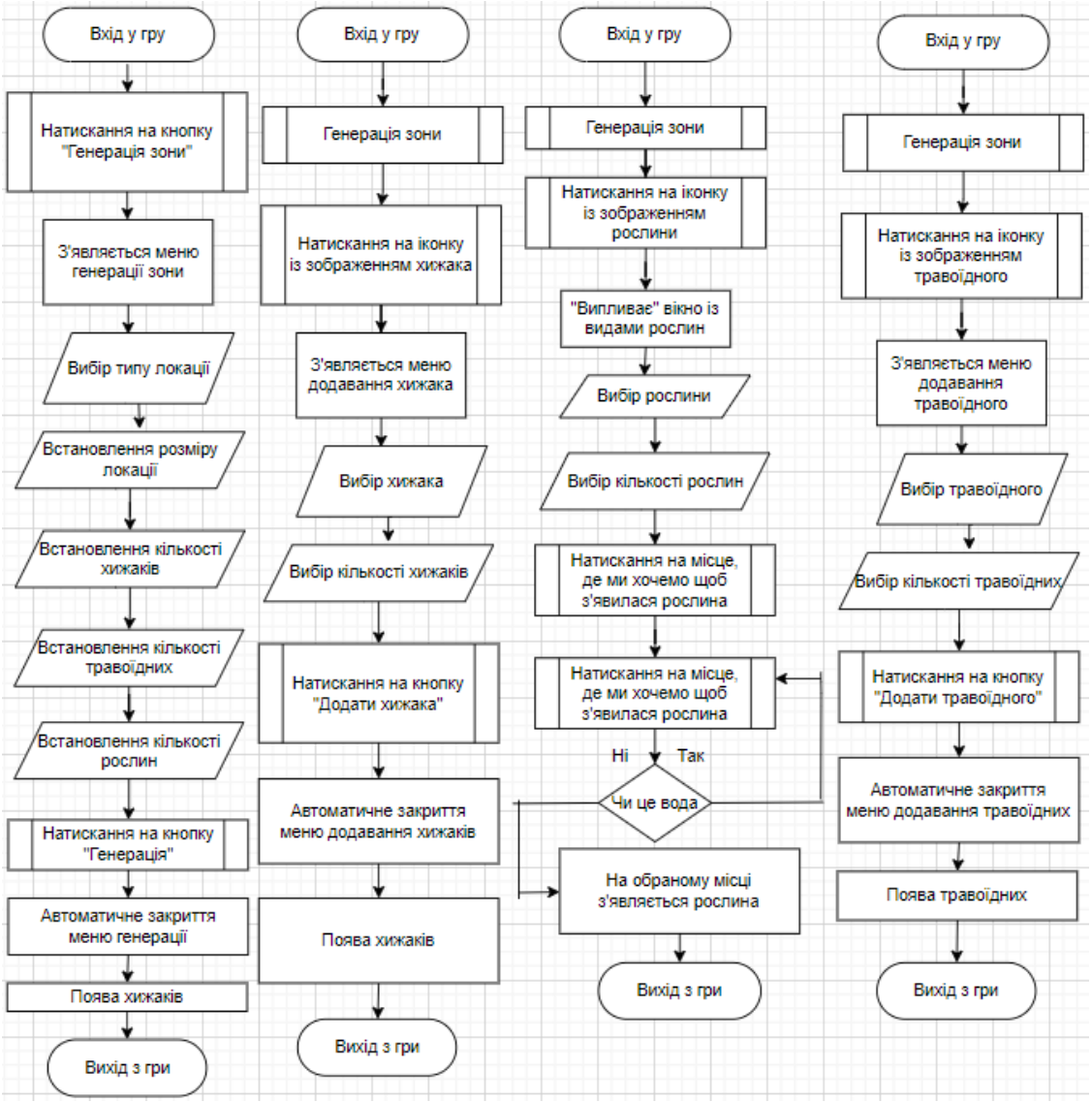

Рисунок 4 – Алгоритми використання для генератору ігрової зони, додавання хижака, травоїдного та рослини

Однією із важливіших типів діаграм була – була діаграма класів. Діаграми класів (Class diagrams) – головний тип діаграм UML, які відображають логічну структуру програмної системи та суттєво впливають на процес генерації програмного коду. Основними елементами діаграми класів є безпосередньо класи (classes) та відношення між ними (relations).

Клас – сукупність логічних об’єктів, які мають схожі характеристики і відрізняються однаковою поведінкою. В ООП характеристики об’єктів певного класу представлені сукупністю атрибутів, а поведінка – сукупністю операцій.

Є п’ять відношень у класах, але в поточному проекті було тільки три відношення, а саме: відношення агрегації, композиції та наслідування.

1. **Відношення агрегації** (aggregation) – відношення типу «частина – ціле», при якому час життя класа частини не співпадає з часом життя класа цілого;
2. **Відношення композиції** (composition) - відношення типу «частина – ціле», при якому час життя класу частини співпадає з часом життя класу цілого;
3. **Відношення наслідування** (generalization) – відношення типу «загальне – часткове»;

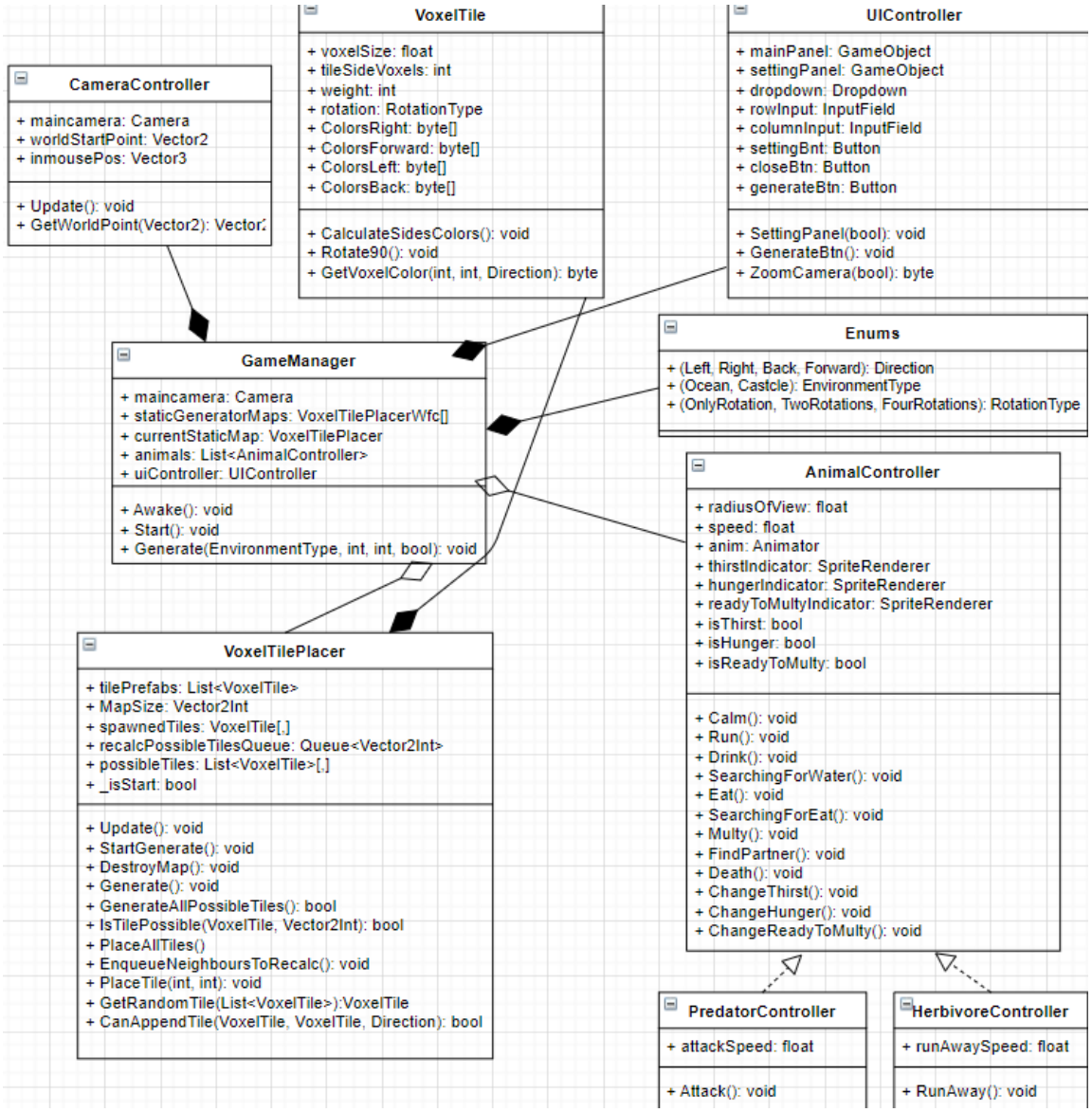

Рисунок 5 – Діаграма класів

Наступний підрозділ був присвячений діаграмі станів та переходів. Діаграма станів відображає скінчений автомат у вигляді графу, вершинами якого є стани об’єкту, поведінка якого моделюється, а переходами – події, які переводять об’єкт, який розглядається, з одного стану в інший. Стан (state) – це

логічна сутність, що використовується для моделювання певної ситуації, дії, процесу. Кожен стан має ім'я та список внутрішніх дій. Список внутрішніх дій містить перелік дій, які виконуються в процесі знаходження системи чи об'єкту в даному стані.

Усього було створено три діаграми станів та переходів. Одна з яких відображає стандартний функціонал.

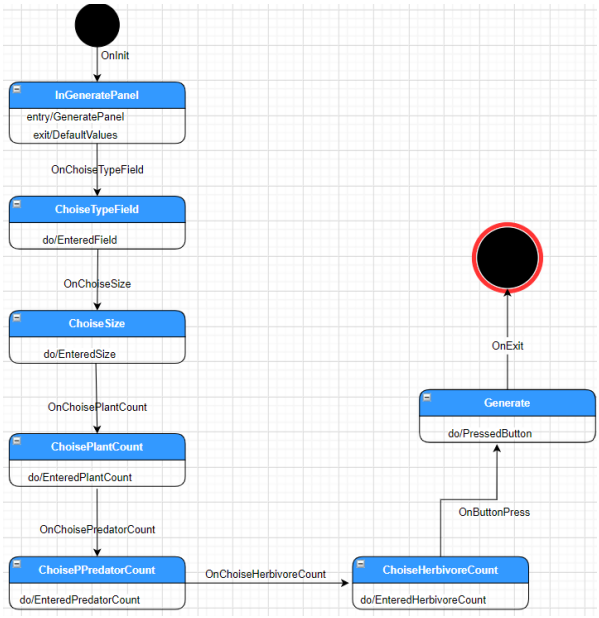

Рисунок 6 – Діаграма станів та переходів «Функціонал для налаштування генерації»

Останнім підпунктом другого розділу було створення макету користувацького інтерфейсу та демонстрація його реалізації у проекті. Одним із таких макетів був присвячений генерації ігрової зони. На рис. 7 можна побачити макет панелі генерації зони.

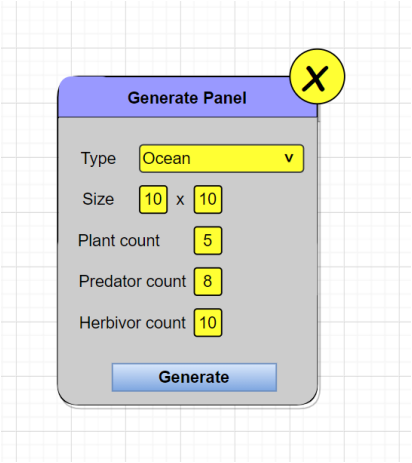

Рисунок 7 – Макет панелі для генерації зони

А тепер на рис. 8 можна побачити фінальну версію користувацького інтерфейсу який стосується генерації зони.

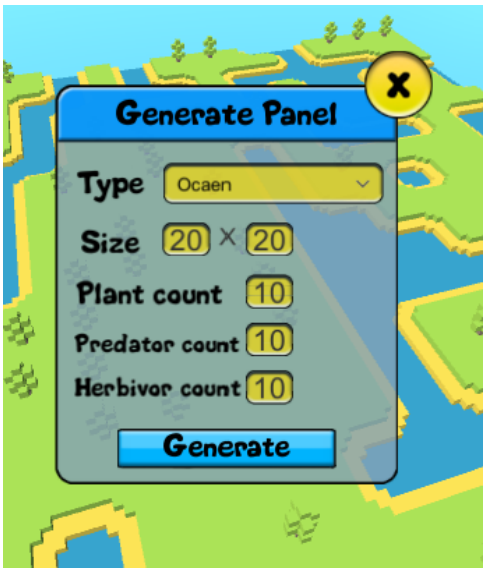

Рисунок 8 – Реалізація панелі генерації у проекті що розробляється

У кінці другого розділу теж є висновок до розділу, в якому також дуже коротко описано все, що було розказано у розділі.

У **третьому** розділі повноцінно розписана розробка програмного застосунку. Але спочатку було досліджено різні ігрові рушії. Кожен із них мав свої переваги та недоліки. З кожним роком їх становиться все більше. Різноманітність полягає у тому, що кожен з них не може бути у всьому кращий за іншого. Один дозволяє розробляти 2D ігри (такі рушіїи як CONSTRUCT 2, Corona, Cocos2D-x) інші 3D (Unreal Engine, Unity 5, CryEngine). Одні безкоштовні, а за використання інших потрібно заплатити місячну підписку, або підписку на рік. У таблиці 3.1 та таблиці 3.2 будуть коротко наведені плюси та мінуси таких рушіїв як: Unity 3D, Unreal Engine, CryEngine та Cocos2D-x.

Таблиця 1 – Плюси та мінуси рушіїв Unity 3D та Unreal Engine

№	Unity3D		Unreal Engine	
	Переваги	Недоліки	Переваги	Недоліки
1	Велике ком’юніті	Погана оптимізація	Вбудоване візуальне програмування	Завищена ціна в магазині на контент
2	Обширна	Дуже нервує коли	Мультизадачність	Мало

	документація	руйнується білд через встановлення різних плагінів, а саме те, що буває не точно описана проблема	та універсальність	універсального контенту
3	Кросплатформість	Часто ігри на телефони багато займають місця	Кросплатформість	Сильно навантажує комп’ютер
4	Низький поріг входження		«Красива картинка» у грі	Великий поріг входження
5	Великий набір ассетів			

Таблиця 2 – Плюси та мінуси рушіїв CryEngine та Cocos2D-x

	CryEngine		Cocos2D-x	
1	Переваги	Недоліки	Переваги	Недоліки
2	У комплекті йде GameSDK (це все що потрібно для шутеру)	Застаріла документація	Безкоштовний	Тільки 2д ігри
3	У нього відмінний realtime render	Маленьке і не найактивніше ком'юніті	Низький поріг входження	
4	Візуальна частина	Дуже скромний магазин ассетів	Швидкий	
		Складний процес складання білда	Декілька мов програмування підтримує	

Коли вибір пав на Unity 3D, потрібно було обрати платформу, на яку буде потім випускатися гра.

Ігрові компанії вибирають декілька тактик: або випускати на дві платформи одночасно, або спочатку на IOS, а потім на Android через деякий час, або тільки на одній якійсь платформі.

У першому випадку, це можливо коли є достатньо коштів на підтримку двох платформ, та бажанню отримувати зразу ж прибуток з двох мобільних гігантів.

Другий спосіб використовують з міркувань безпеки самого ігрового продукту. На приклад, в грі є внутрішні покупки, де за реальну валюту можна придбати щось, що може полегшити ігровий процес. Зважаючи на те, що відсоток зламування ігор на IOS набагато менше ніж на Android, тому вигідніше випускати спочатку саме на IOS, бо розробники зможуть отримати великий прибуток, а потім вже на Android, коли ігровий продукт вже окупився. А от з Android все печально, будь яка гра може бути зламана, та на неї можуть встановити моди, які дозволять купляти предмети безкоштовно, це дуже сильно може вдарити по успіху тієї компанії, що створила гру. В нашому випадку ведеться розробка одночасно для IOS та Android.

Після цього, в наступному підрозділі вже почалася фаза розробки, але спочатку пояснення як користуватись Unity 3d, які він має вікна, які в ньго є компоненти та що повни означаюсь.

Далі була описана розробка алгоритму для процедурної генерації ігрової локації. Процедурна генерація буде створюватись за допомогою правильної підстановки «тайлів». Ці тайли представляюсь собою квадратні шматки поверхні, які містять в собі якийсь рельєф, на приклад зелена земля, доріжка коричнева, чи вода блакитного кольору. Ці тайли будуть робитись в програмі MagicaVoxel.

MagicaVoxel – програма для створювання вексельних 3Д моделей. Ця програма безкоштовна та дуже проста у використанні. Саме тому, для швидкості та якості було обрано саме цю програму.

Останній розділ був присвячений розробці штучного інтелекту для тварин, та закінченню розробки усього проекту.

ЗАГАЛЬНІ ВИСНОВКИ

Підчас виконання кваліфікаційної роботи було опрацьовано багато матеріалів в області штучного інтелекту. Були розглянуті аналоги ігор, в яких використовувався штучний інтелект тварин. Також були розглянуті ігри, які мають схожий функціонал, схожу ідею та реалізацію. В процесі виконання проекту, із цих ігор були взято найкраще, та усунено недоліки.

Щоб гра не була одноманітною, було вирішено створити процедурну генерацію ігрової локації. Її можна налаштовувати та обирати її стиль.

Було досліджено вплив інтеграції відеоігор та застосунків в якості навчального матеріалу у школах. Ці дослідження показали, що все частіше у навчальних закладах починають інтегрувати відеоігри та застосунки для покращення якості засвоєння матеріалу.

Було створено специфікацію вимог до розробляє мого програмного забезпечення. Встановлений основний функціонал гри та вимоги до його використання.

Окремий розділ був присвячений проєктуванню програмного забезпечення. Це один із найважливіших моментів розробки програмного забезпечення. В ньому створюються основні діаграми та опис використання. Це допомагає уявити як програмне забезпечення буде виглядати у кінці розробки. За цими діаграмами легко розумієш послідовність розробки проєкту. Також в голові не потрібно буде тримати весь функціонал, він завжди буде перед очима. Були розглянуті такі діаграми як: діаграми взаємодії, діаграми використання, діаграми станів та переходів, діаграма класів та алгоритми використання. Окремий підрозділ був присвячений створенню макету майбутнього користувацького інтерфейсу.

Перш ніж почати створювати продукт, потрібно було обрати платформу на якій буде працювати розробляє мий продукт, вибір пав на ОС Android.

Не менш важливим було обрати середовище розробки. Серед чотирьох претендентів, було обрано ігровий рушій Unity 3D. Він простіше, але в той же час має велику кількість функціоналу. Завдяки кваліфікаційній роботі було здобуто велику кількість навичок у роботі із ігровим рушієм Unity 3D.

В основній частині розробки був детально описаний алгоритм створення ігрової зони завдяки процедурній генерації. Це було б не можливо без використання такої програми для вексельного моделювання як MagicaVoxel.

Останнім етапом було створення штучного інтелекту для тварин, створення імітації екосистеми у дикій природі.

Поставлені задачі були виконані, мета кваліфікаційної роботи була досягнута.

АНОТАЦІЯ

До кваліфікаційної роботи бакалавра

«Мобільна гра-симулятор екосистеми дикої природи з використанням штучного інтелекту»

Студент 408 гр.: Чимбір Сергій Миколайович

Керівник: канд. техн. наук, доцент (б.в.з.) кафедри ІПЗ Горбань Гліб Валентинович

Метою дипломної є створення імітації штучного інтелекту тварин для наочної демонстрації процесів в екосистемі дикої природи. Для цього потрібно дослідити поведінку травоїдних та хижаків, та відносно цих даних створити штучний інтелект, який зможе відтворити їх поведінку та зробити ілюзію справжньої екосистеми у дикій природі. Робота складається із чотирьох розділів.

У першому розділі описано аналіз предметної сфери, процедурна генерація та специфікація вимог.

У другому розділі описане проєктування програмного застосунку.

У третьому розділі описана реалізація програмного застосунку.

Окремо виділена спеціальна частина яка стосується охорони праці.

Результатом роботи є гра на платформу Android, яка, завдяки штучному інтелекту та процедурній генерації локації зміг відтворити екосистему у дикій природі. Дана гра призначена для використання в освітніх цілях, зокрема для використання у школах та університетах для більш інтерактивної взаємодії учнів та навчального матеріалу.

Ключові слова: штучний інтелект, процедурна генерація, екосистема, Unity 3D, Android, тварини, C#, MogicaVoxel.

ABSTRACT

of the Bachelor’s Thesis

"Mobile game-simulator of wildlife ecosystem using artificial intelligence"

Student of group 408 .: Chimbir Sergii Mykolaiovich

PhD, Associate Professor of Department of Software Engineering Horban Hlib Valentynovych

The purpose of the diploma is to create an imitation of artificial intelligence of animals for visual demonstration of processes in the wildlife ecosystem. To do this, we need to study the behavior of herbivores and predators, and in relation to this data to create artificial intelligence that can reproduce their behavior and create the illusion of a real ecosystem in the wild. The work consists of four sections.

The first section describes the analysis of the subject area, procedural generation and specification of requirements.

The second section describes the design of the software application.

The third section describes the implementation of the software application.

There is a special section on labor protection.

The result is a game on the Android platform, which, thanks to artificial intelligence and procedural location generation, was able to recreate the ecosystem in the wild. This game is intended for educational purposes, in particular for use in schools and universities for more interactive interaction between students and educational material.

Keywords: artificial intelligence, procedural generation, ecosystem, Unity 3D, Android, animals, C #, MogicaVoxel.