КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
БУДІВНИЦТВА І АРХІТЕКТУРИ

автоматизації і інформаційних технологій
(факультет)

інформаційних технологій проєктування та прикладної математики
(кафедра)

ПОЯСНЮВАЛЬНА ЗАПИСКА
ДО КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА ЗДОБУТТЯ ОСВІТНЬОГО РІВНЯ «БАКАЛАВР»

на тему: «Інформаційна система для створення та оцінки якості відеогри»

ТЕЛЕНЧЕНКО РОСТИСЛАВ СЕРГІЙОВИЧ
(прізвище, ім’я та по батькові студента повністю)

Київ 2026 р.

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
БУДІВНИЦТВА І АРХІТЕКТУРИ

автоматизації і інформаційних технологій
(факультет)

інформаційних технологій проєктування та прикладної математики
(кафедра)

ЗАТВЕРДЖУЮ
Завідувач кафедри ІТППМ
д.т.н., професор Бородавка Є.В.

_____
“___” _____2026 року

ПОЯСНЮВАЛЬНА ЗАПИСКА
ДО КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА ЗДОБУТТЯ ОСВІТНЬОГО РІВНЯ «БАКАЛАВР»

на тему: «Інформаційна система для створення та оцінки якості відеогри»

Виконав: студент 4-го курсу, групи ІСТ-22

Спеціальності: 126 «Інформаційні системи та технології»
(шифр і назва спеціальності)

Теленченко Р.С.
(прізвище та ініціали)

Керівник доктор філософії Бугров А.А.
(прізвище та ініціали)

Рецензент доктор філософії Рябчун Ю.В.
(прізвище та ініціали)

Київ 2026 р.

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
БУДІВНИЦТВА І АРХІТЕКТУРИ

Факультет: автоматизації і інформаційних технологій

Кафедра: інформаційних технологій проєктування та ПМ

Освітній рівень: «бакалавр» за ОПП

Спеціальність: 126 «Інформаційні системи та технології»

Освітня програма: «Інформаційні системи та технології»

ЗАТВЕРДЖУЮ

Завідувач кафедри ІТППМ

д.т.н., професор Бородавка Є.В.

_____

“___” _____2026 року

З А В Д А Н Н Я

ДО КВАЛІФІКАЦІЙНОЇ РОБОТИ

НА ЗДОБУТТЯ ОСВІТНЬОГО РІВНЯ «БАКАЛАВР»

Теленченко Ростислав Сергійович
(прізвище, ім'я, по-батькові)

1. Тема роботи: “Інформаційна система для створення та оцінки якості відеогри”
затверджена наказом ректора КНУБА № 444/52-15/13.6/26 від “08” квітня 2026 року
2. Керівник роботи: Бугров Анатолій Анатолійович, доктор філософії, старший викладач кафедри ІТППМ
3. Строк подання студентом роботи до захисту червень 2026 року
4. Зміст пояснювальної записки за розділами:
   Р. 1. Аналіз предметної області та постановка задачі
   Р. 2. Проектування архітектури системи
   Р. 3. Розробка відеогри на базі інформаційної системи
   Р. 4. Система оцінки якості відеогри за критеріями
5. Інформаційні слайди:
   С. 1. Титульний слайд
   С. 2. Мета роботи
   С. 3. Актуальність роботи

С. 4. Задачі та вимоги
С. 7. Система процесу створення
С. 8. Дерево функцій
С. 12. Приклад використання системи
С. 16 Оцінка за критеріями
С. 20. Висновки

6. Календарний план виконання робіт:

Види робіт та їх зміст	Дата виконання
Розділ 1. Аналіз предметної області та постановка задачі	Лютий 2026 р.
Розділ 2. Проектування архітектури системи	Березень 2026 р.
Розділ 3. Розробка відеогри на базі інформаційної системи	Квітень 2026 р.
Розділ 4. Система оцінки якості відеогри за критеріями	Травень 2026 р.
Остаточне оформлення роботи	Червень 2026 р.
Направлення роботи на рецензування	Червень 2026 р.
Попередній захист роботи на кафедрі	Червень 2026 р.

7. Консультанти розділів кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада консультанта, представника комісії	Дата	Підпис
Прийом програмного продукту	Доктор філософії, доцент Рябчун Ю.В.		

8. Дата видачі завдання: 12 січня 2026 року

Керівник _____ Бугров А.А.
(підпис) (прізвище та ініціали)

Бакалавр _____ Теленченко Р.С.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

«Інформаційна система для створення та оцінки якості відеогри».

Кваліфікаційна робота бакалавра за спеціальністю: 126 «Інформаційні системи та технології», освітня програма: «Інформаційні системи та технології». – Київський національний університет будівництва та архітектури. – Київ, 2026.

Робота присвячена покращенню процесів взаємодії та умов праці робітників в індустрії розробки відеоігор шляхом створенням та опису системи взаємодії її компонентів між собою.

В результаті роботи було створено та застосовано систему процесу створення та оцінки якості відеогри.

Ключові слова : Інформаційна система, розробка відеогри, оцінка якості відеогри.

SUMMARY

«Information system of the process of creating and evaluating the quality of a video game».

Bachelor's qualification work in the specialty: 126 "Information systems and technologies", educational program: "Information systems and technologies". - Kyiv National University of Construction and Architecture. - Kyiv, 2026.

The work is devoted to improving the interaction processes and working conditions of workers in the video game development industry by creating and describing a system of interaction between its components.

As a result of the work, a system of the process of creating and evaluating the quality of a video game was created and applied.

Keywords: Information system, video game development, video game quality assessment.

ЗМІСТ

ВСТУП

Робота присвячена покращенню процесів взаємодії та умов праці робітників в індустрії розробки відеоігор шляхом створенням та опису системи взаємодії її компонентів між собою. Та підвищення якості відеоігор за допомогою системи оцінки якості.

Актуальність задачі. Індустрія створення відеоігор є одним з найбільших сегментів індустрії розваг. Масштаби ігрової індустрії можна порівняти, наприклад, з кіноіндустрією. А по швидкості росту по за останні роки індустрія відеоігор істотно її випереджала.

Проблема, яку необхідно вирішити з позиції системного аналізу. Це досить молода індустрія і її методи роботи знаходяться на етапі активної розробки і є досить не досконалими, що створює багато проблем в процесі розробки відеогри. Серед них такі як організація робочого процесу, переробка та розподіл задач. Розглянуто, що сповільнює темп та підвищує витрати розробки проекту.

В даній дипломній роботі досліджується принципи функціонування індустрії відеоігор. Розробляються методи взаємодії її компонентів. Створюється система оцінки якості відеогри за критерія, що допоможе підвищити рівень створюваної продукції.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Опис предметної області

Предметна область даної дипломної роботи це – індустрія комп'ютерних ігор. Це – сектор економіки, пов'язаний з розробкою, просуванням і розповсюдженням комп'ютерних ігор.

Відеогра – це електронна гра, в ігровому процесі якої гравець використовує інтерфейс користувача, щоб отримати зворотну інформацію з пристрою. Електронні пристрої, які використовуються для того щоб грати, називаються ігровими платформами.

Індустрія комп'ютерних ігор зародилася в середині 1970-х років як рух ентузіастів і за кілька десятиліть виросла з невеликого ринку в індустрію світовий обсяг якої оцінюється в 151,06 млрд. доларів США у 2019 році та в 162,32 млрд. доларів США у 2020 році, і очікується, що він буде продовжувати зростати оскільки ігри зараз забезпечують один із найбільш захоплюючих та вражаючих видів розваги для більш ніж двох мільярдів людей по всьому світу.

Сучасні персональні комп'ютери безліччю нововведень зобов'язані саме ігрової індустрії. До числа найбільш значущих відносять звукові і графічні карти та центральні процесори. Звукові карти спочатку були розроблені для інтегрування якісного цифрового звуку в комп'ютерні ігри, і тільки потім звукове обладнання було вдосконалено під потреби меломанів. Графічні карти, які на зорі комп'ютерної ери еволюціонували в напрямку збільшення кількості підтримуваних квітів, пізніше стали розвиватися для апаратної підтримки графічних інтерфейсів користувача та ігор. Для GUI потрібно збільшення дозволу екрану, а для ігор прискорення 3D графіки.

Сучасні комп'ютерні ігри – одні з найбільш вимогливих додатків на ПК. Потужні комп'ютери купуються геймерами, які потрібні для запуску новітніх ігор, в яких використовуються самі передові технології. Таким

чином, ігрова індустрія тісно пов'язана з індустрією виробництва центральних процесорів і інших компонентів ПК, так як відеоігри часто вимагають більш високих апаратних потужностей, ніж бізнес-додатки.

Створення гри це тривалий і трудомісткий процес, що складається з найрізноманітніших етапів, що включає в себе як технічні, так і творчі моменти. Ось тому, в більшості своїй, ігри створюють не окремі особистості, а цілі команди розробників. Кожна окрема людина в команді - фахівець у своїй галузі знань.

1.1.1 Опис спеціальностей залучених до індустрії

В індустрію комп'ютерних ігор входить велика кількість спеціальностей, за якими працюють десятки тисяч людей по всьому світу.

Основні ролі і посади в розробці відеогри:

- Геймдизайнер — придумує концепцію гри, її основні риси. Він проектує базову ігрову механіку, ігровий процес, образи персонажів, коригує їх впродовж розробки. Як правило гейм дизайнер вже має досвід роботи з відеоіграми на інших посадах. Може бути кілька гейм дизайнерів, з-поміж яких один — провідний дизайнер, головний, а інші відповідають за дизайн окремих аспектів: персонажів, рівнів, механіки.
- Менеджер — координує роботу людей, залучених до створення гри. Розподіляє обов'язки, встановлює строки виконання кожного виду робіт. Планує витрати. Серед менеджерів існує спеціалізація на генерального директора, головного менеджера і продюсера.
- Програміст — пише і вдосконалює програмний код гри. Створює базову ігрову механіку, файлову архітектуру проекту. Можливий розподіл праці: технічний директор, програміст-проектувальник, програміст інтерфейсу, графічного рушія і т. д.
- Художник — створює заставки, спрайти, текстури, тривимірні моделі, оформлення інтерфейсу. Часто існує розподіл, де один художник

займається текстурами, інший — спецефектами, анімацією і т. д. Головний художник, який координує роботу інших, зветься арт-директором.

- Звукоінженер — знаходить і записує звукові ефекти чи синтезує їх.
- Сценарист — складає сценарій гри, загальний сюжет і окремих подій. Пише тексти діалогів, описів предметів і персонажів.
- Тестер — грає в різні версії гри, випробовує її роботу в різних ситуаціях, виявляє та документує помилки і недоліки. Для тестування існують спеціальні відділи, розробника і видавця гри, незалежні тестери, і тестери з числа звичайних користувачів.

1.1.2 Опис етапів створення відеогри

Процес створення гри або конвеєр розробки гри - це процес створення відеоігри від концепції до завершення. Подібно виробничої лінії, конвеєр розробки ігор допомагає організувати потік роботи, щоб кожен знав, що і коли потрібно реалізувати. Конвеєр також допомагає керувати графіком розробки ігор і бюджетом, зменшуючи неефективність і вузькі місця.

Хоча конвеєри розрізняються між проектами і студіями, процес досить схожий незалежно від того, чи працюєте ви над інді або мобільного грою.

Гра постійно розвивається, і речі, які здавалися прекрасними в теорії, можуть не працювати так добре в реальності. Отже, конвеєр не обов'язково є лінійним процесом. Робота повинна бути спрямована на творче схвалення і часто може повернутися назад для доопрацювання. Конвеєри повинні бути досить гнучкими, щоб враховувати перегляди і зміни курсу.

Розробка відеоігор зазвичай ділиться на 3 етапи: пре-продакшн, продакшн і пост-продакшн.

1. Підготовка до виробництва

Тут починається кожен проект. По суті, Препродакшн визначає, про що гра, навіщо її робити і що потрібно для її створення. У вас може бути

відмінна ідея для типу гри, історії, яку ви хочете втілити в життя, або ви можете захотіти створити таку, яка використовує певний тип технологій (наприклад, VR, новий контролер або консоль). На етапі підготовки до виробництва ви повинні мати відповіді на такі питання, як:

- Про що гра?
- Хто наша цільова аудиторія / користувачі / публіка?
- Чи є аналоги на цьому ринку? Яка конкуренція?
- На якій платформі буде створено проект?
- Як це буде монетизувати? Чи буде він продаватися на платформі або буде продаватися безкоштовно з внутрішньо ігровими покупками?
- Скільки часу буде потрібно на розробку?
- Який персонал і ресурси для цього будуть потрібні?
- Який орієнтовний бюджет?

Цей етап може тривати від тижня до року, в залежності від типу проекту, наявних ресурсів і фінансів, і зазвичай займає до 20% від загального часу виробництва.

Прототипування

Прототип відеоігри - це необроблений тест, який перевіряє функціональність, призначений для користувача досвід, ігровий процес, механіку і художнє оформлення. Прототипування відбувається на етапі підготовки до виробництва, щоб перевірити, чи буде ідея гри працювати і чи варто її реалізовувати. Багато ідей не проходять цю стадію.

Команда часто починає з паперових проектів, щоб перевірити теорії і пропрацювати багато нюансів гри або ряду систем швидко, легко і економічно.

Мета полягає в тому, щоб якомога швидше запустити прототип і перевірити, чи дійсно ваші ідеї працюють і наскільки цікава гра. Прототипування також може виявити несподівані проблеми, які потенційно

можуть змінити весь хід проекту. Важливо, щоб інші тестували ваш прототип, тому що речі, які очевидні для вас, можуть не бути для інших.

Інформація, зібрана на цьому етапі підготовки виробництва, становить основу документа по дизайну гри. Документ ігрового дизайну (GDD) - це по суті північна зірка гри. Це живий документ, який допомагає кожному зрозуміти і прийняти більш широке бачення проекту.

GDD включає в себе:

- Схема гри. Що повинен робити гравець, яка кінцева мета, що заважає її досягненню.
- Інтерфейс. Детально описана функціональна частина (що можна робити, яким чином - меню, миша, гарячі клавіші, кнопки ...).
- Ігрова механіка. Як влаштований ігровий світ, які характеристики є у його об'єктів, формули руху, бою і всього іншого, рольова система, фізика.
- Програмні механізми і алгоритми. Які характеристики матимуть графічний движок, II, мережевий код, інтерфейс, редактор карт, звук.
- Графіка. Скільки і яких вам знадобиться моделей, анімацій, двовимірної графіки, роликів, шпалер (так, і їх теж варто запланувати заздалегідь). Тут вкрай бажані (може, краще сказати - необхідні) хоч якісь начерки, concept art, за якими можна відчути візуальний стиль гри.
- Звуки і музика. Теми, вид і спосіб відображення звуків, набір звукових ефектів.
- Сюжет. Загальна сюжетна компанія, план кампаній, основні завдання і т. П. - в залежності від жанру. Кожна з передбачуваних карт повинна бути запланована тут.

- Ігровий світ. Основні персонажі / монстри / види військ з параметрами і зразковим розташуванням / способом видобутку і виробництва.
- Працівники. зарплати, терміни і план роботи.

Як живий документ, GDD постійно оновлюється і вдосконалюється в процесі виробництва. Це може бути пов'язано з технічними або фінансовими обмеженнями або просто з усвідомленням того, що деякі речі не виглядають, не працюють і не працюють так добре, як ви спочатку сподівалися.

Багато людей, особливо дрібні розробники, люблять використовувати більш гнучкі методи розробки, які менше пов'язані з процесом і документацією, а більше з простими побудовами. Однак більші студії надають перевагу іншому підходу. EA, Microsoft, Sony, Ubisoft та інші великі ігрові компанії сильно орієнтовані на процеси і вимагають складної документації. Це велика частина того, як вони досягають успіху знову і знову.

GDD підтримує організованість, допомагає виявляти потенційні ризики і дозволяє заздалегідь побачити, кого, можливо, доведеться найняти / передати на аутсорсинг, щоб втілити проект в життя.

2. Виробництво

Виробництво - це найдовший етап розробки відеогри. Практично все в відеоіграх - це усвідомлене рішення. Сюди входять всі рішення про вигляд персонажів, оточення, об'єктів, а також кольори, звуки, рівень складності, правила і система нарахування балів та інше. Однак початкові ідеї не завжди так добре втілюються в життя, тому в процесі розробки гра постійно тестується і допрацьовується.

Основні етапи виробництва

- Прототип: це початкове випробування гри (яке відбувається на стадії підготовки до виробництва і детально описано вище).

- Перша тест: перша тест дає набагато краще уявлення про зовнішній вигляд та ігровий процес. Хоча він ще далекий від фіналу, контент замінюються більш якісними активами і додаються ілюстрації.
- Вертикальний фрагмент: вертикальний фрагмент - це повністю відтворений зразок, який можна використовувати для презентації вашої гри студіям або інвесторам. Вертикальний зріз, від декількох хвилин до півгодини, дозволяє побачити гру з перших рук.
- Пре-альфа: більша частина контенту розробляється на етапі пре-альфа. На цьому етапі розробки гри потрібно буде прийняти кілька важливих рішень. Контент може бути вирізаний, або для поліпшення ігрового процесу буде потрібно додати нові елементи.
- Альфа: гра «завершена», що означає, що всі основні функції були додані, і в гру можна грати повністю від початку до кінця. Деякі елементи, такі як художні активи, все ж, можливо, буде потрібно додати, але елементи управління і функції повинні працювати правильно. Тестувальники стежитимуть за тим, щоб все працювало без збоїв, і повідомляли про помилки команді.
- Бета: на цьому етапі весь контент і ресурси інтегровані, і команді слід зосередитися на оптимізації, а не на додаванні нових функцій або можливостей.
- Gold master: гра остаточна і готова до відправки в видавничий центр і випуску для широкої публіки.

3. Постпродакшн

Після завершення виробництва і випуску гри процес розробки гри триває, і деякі члени команди переводяться на обслуговування (виправлення помилок, створення патчів) або створення бонусного або завантаження контенту (DLC). Інші можуть перейти до наступного проекту.

Може бути проведено опитування, щоб обговорити, що спрацювало, а що не спрацювало, і визначити, що можна було б зробити краще наступного

разу. Всі проектні документи, активи і код доопрацьовуються, збираються і зберігаються на випадок, якщо вони знадобляться в майбутньому.

1.2 Аналіз та постановка задачі

Інформаційні системи (ІС) – сукупність організаційних і технічних засобів для збереження та обробки інформації з метою забезпечення інформаційних потреб користувачів. Вони відрізняються за типами об’єктів управління, характером та обсягом розв’язуваних задач та ін.

Проектування ІС – процес, спрямований на вдосконалення економічної інформації системи об’єкта управління (ОУ), що передбачає створення та впровадження комплексного розв’язання економічних задач із застосуванням сучасних електронних обчислювальних машин (ЕОМ) і технічних засобів управління об’єктом.

Для успішної реалізації проекту об'єкт проектування повинен бути насамперед адекватно описаний, побудовані повні і несуперечливі функціональні та інформаційні моделі ІС. Накопичений на даний час досвід проектування ІС показує, що це складна, трудомістка і тривала за часом робота, що вимагає високої кваліфікації фахівців, які беруть участь у ній. Однак донедавна проектування ІС виконувалося в-основному на інтуїтивному рівні з застосуванням неформалізованих методів, які базуються на мистецтві, практичному досвіді, експертних оцінках і дорогих експериментальних перевірках якості функціонування ІС. Крім того, у процесі створення і функціонування ІС інформаційні потреби користувачів можуть змінюватися або уточнювати, що ще більше ускладнює розробку і супровід таких систем.

Система створюється для:

- Підвищення ефективності створення відеогри;
- Покращення рівня якості відеоігор;

Вимоги для системи:

- Повинні бути описані входи та виходи системи;
- Створити моделі взаємодії системи;
- Визначені функції системи;
- Визначений порядок виконання цих функцій;
- Повинна містити інструкції по процесу оцінки якості;

1.3 Вибір технології реалізації

Unity - це крос-платформний ігровий движок, розроблений Unity Technologies, вперше оголошений і випущений у червні 2005 року на всесвітній конференції розробників Apple Inc. як ексклюзивний ігровий движок для Mac OS X. З тих пір поступово розширюється для підтримки різноманітних настільних, мобільних, консольних та віртуальних платформ.

Unity - це один з найпопулярніший у світі ігрових движків. Він поєднує в собі багато різних функцій і досить гнучкий, щоб зробити майже будь-яку гру, яку ви можете собі уявити. Завдяки неперевершеним крос-платформним функціям Unity користується популярністю як серед розробників початківців, так і серед професійних студій.

Програмне забезпечення Unity - це IDE. IDE розшифровується як «інтегроване середовище розробки», що описує інтерфейс, який надає безліч інших корисних функцій та інструментів, необхідних для розробки, в одному місці (рис. 1.1). Як ігровий движок, Unity здатний надати багато найважливіших вбудованих функцій, які змушують гру працювати. Це означає такі речі, як фізика, 3D-рендерінг та виявлення зіткнень. З точки зору розробника, це означає, що немає необхідності винаходити колесо заново. Замість того, щоб розпочинати новий проект, створюючи новий фізичний двигун з нуля - обчислюючи кожен останній рух кожного матеріалу, або те, як світло має відбиватися від різних поверхонь.

Unity 3D постачається з безліччю професійних інструментів як для програмістів, так і для художників. Unity надає робочий простір, який

поєднує в собі інструменти, зручні для виконавців, та дизайн, керований компонентами, що робить розробку гри досить по-інтуїтивному зрозумілою.

Розробка 2D та 3D можливо в Unity, а фізика 2D обробляється популярним двигуном Box2D. Unity використовує компонентний підхід до розробки ігор, що обертається навколо збірних плат. За допомогою збірних панелей дизайнери ігор можуть ефективніше будувати об’єкти та середовища та швидше масштабуватись.

Завдяки потужним шейдерам, матеріалам, заснованим на фізиці, системам пост-обробки та системам освітлення з високою роздільною здатністю, Unity може надати Міжплатформене розгортання - це основна привабливість для сучасних розробників, і Unity є лідером в цій галузі. Завдяки підтримці всіх основних консолей та операційних систем, ігри, розроблені в Unity, можна розгорнути на абсолютно будь-якій платформі.

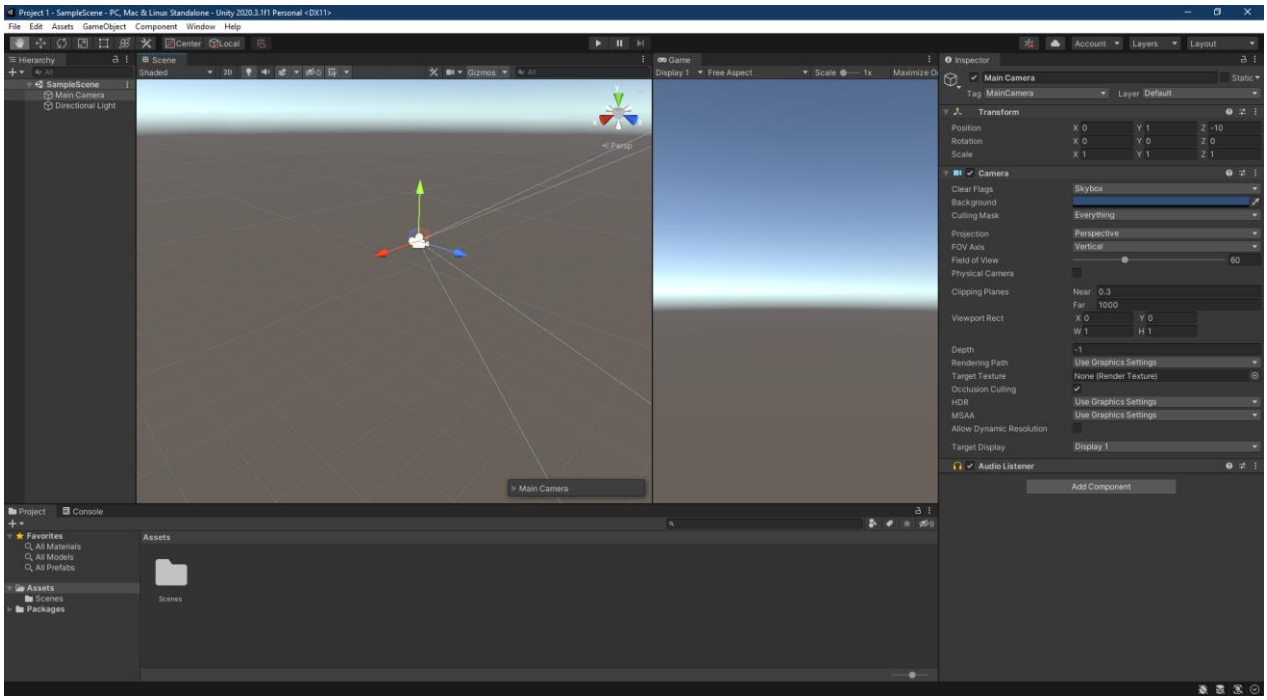

Рисунок 1.1 Інтерфейс програми Unity

За допомогою інструментів редактора Unity ви можете одночасно обробляти дані для мишей, клавіатур та ігрових контролерів.

Існує також досить потужна підтримка хмарних рішень для багатокористувацьких ігор із хостингом на сервері та масштабованим

встановленням зв’язків, що робить його універсальним рішенням для мультиплікаційного досвіду.

Співпраця команд значно покращилася в нових версіях Unity. Вбудований контроль версій та хмарна інтеграція полегшують роботу з іншими як ніколи раніше.

А Unity має настроюваний редактор з повною підтримкою API для створення власних інструментів та сценаріїв редактора. Зробіть майже будь-який інструмент, який ви хочете мати для Unity, за допомогою Unity.

Також варто згадати про магазин асетів, який містить тисячі моделей, сценаріїв, сцен, матеріалів та всього іншого, що ви могли б побажати. Ви навіть можете продати власні активи в магазині Unity.

Але головною причиною вибору Unity є величезна бібліотека ресурсів, доступних кожному. Навіть досвідчені розробники можуть заощадити час і багато чому навчитися у спільноти.

Unity також пропонує надійний набір хмарних інструментів, які дозволяють легко монетизувати вашу гру та додавати багатокористувацькі можливості. Завдяки Unity Analytics, Unity Ads, Unity Collaborate і Unity Multiplayer користувачі мають доступ до неймовірно сумісного набору інструментів для створення динамічних ігор - все в одному місці. Дуже мало інших ігрових двигунів пропонують таку централізацію.

У додатку Unity підтримується ряд мов програмування, зокрема JavaScript і C #. В даній роботі буде використовуватися C #.

C# — об'єктно-орієнтована мова програмування з безпечною системою типізації для платформи .NET [6].

Особливості, якими володіє C #:

- **компільована мова**. У компільованій мові, такій як C #, процес програмування включає три окремих етапи:
    1. Програміст пише програму на мові розробки, такому як C #;
    2. Компілятор перетворює код на мові розробки в код на машинній мові для комп'ютера конкретного типу;

3. Комп'ютер виконує скомпільовану програму.

Тут додається проміжний етап компіляції, в ході якого код на мові розробки перетворюється в виконуваний код (тобто в додаток), який комп'ютер може виконувати безпосередньо, без проміжної інтерпретації. Оскільки компілятор має повне уявлення про програму і платформі, на якої вона повинна виконуватися, він може застосовувати різну оптимізацію. В іграх ці оптимізації дозволяють отримати більш високу частоту кадрів, використовувати більш детальну графіку і забезпечити більш швидку реакцію на дії користувача. Більшість високо-бюджетних ігор пишеться на компільованих мовах, що дає високу швидкість за рахунок оптимізації, але це також означає, що для кожної платформи доводиться компілювати різний виконуваний код.

- **керований код.** C# — керована мова, тобто виділення та звільнення пам'яті в ньому виконується автоматично. Виділення пам'яті - це процедура резервування деякого обсягу в оперативної пам'яті комп'ютера для зберігання даних. Оперативної пам'яті працює набагато швидше жорсткого диска, тому всі програми прагнуть дістати свої ресурси, такі як зображення і звуки, з жорсткого диска і розмістити їх в ОЗУ, щоб мати до них швидший доступ.
- **сувора підтримка типів.** C# — строго типізована мова. Це означає, що при створенні змінної ви повинні повідомити тип даних, які вона зможе зберігати. Може здатися, що сувора типізація ускладнює програмування, але в дійсності вона дає компілятору можливість застосовувати різні оптимізації та дозволяє середовищі розробки MonoDevelop перевіряти синтаксис в режимі реального часу.
- **заснований на функціях.** Створення процедурних мов тобто мов, які використовують функції дозволило програмістам створювати іменовані фрагменти коду і укладати в них певну функціональність тобто групувати послідовності дій під одним ім'ям функції.

- **об'єктно-орієнтована.** В основі концепції об'єктно-орієнтованого програмування лежить поняття об'єкта — певної сутності, яка об'єднує в собі поля (дані) і методи (виконувані об'єктом дії).

  В сучасних ОО-мовах використовуються механізми:

  1. Успадкування. Створення нового класу об'єктів шляхом додавання нових елементів (методів) до вже наявного. Деякі ОО-мови дозволяють виконувати множинне успадкування, тобто об'єднувати в одному класі можливості кількох інших класів.
  2. Інкапсуляція. Приховування деталей реалізації, яке дозволяє вносити зміни в частини програми безболісно для інших її частин, що істотно спрощує супровід і модифікацію ПЗ.
  3. Поліморфізм. За поліморфізму деякі частини (методи) батьківського класу замінюються новими, що реалізують специфічні для даного нащадка дії. Таким чином, інтерфейс класів залишається колишнім, а реалізація методів з однаковою назвою та набором параметрів різниться.

2 ПРОЕКТУВАННЯ АРХІТЕКТУРИ СИСТЕМИ

2.1 Цільовий аналіз розробки

Метою створення нашої системи є покращення процесу розробки відеогри, а саме створення моделей взаємодії людей залучених до її створення. Але для початку необхідно звернути увагу як система взаємодіє з зовнішнім середовищем. Зовнішнє середовище впливає на систему через ресурсне забезпечення, керування і різного роду контрольовані і неконтрольовані фактори, що сприяють або перешкоджають нормальному функціонуванню системи.

Такі зв'язки зовнішнього середовища із системою (дія середовища на систему) називають входами системи. А цільові продукти системи - це виходи системи.

Продемонструємо вхідні та вихідні елементи нашої системи за допомогою рис. 2.1 контекстної моделі чорної скриньки, щоб продемонструвати цілі роботи.

Рисунок 2.1 Контекстна модель ’’Чорна скринька’’

- Верхня сторона має значення "Управління" (Control);
  У нашому випадку це МАІ;
- Ліва сторона має значення "Вхід" (Input);
  На вхід подаються такі дані як: цілі, ідеї для розробки, людські та інші ресурси;
- Права сторона має значення "Вихід" (Output);
  На вихід передається така інформація: готова відеогра та проектна документація;
- Нижня сторона має значення "Механізм" (Mechanism);
  З низу на скриньку впливають такі механізми як: ОПР.

Отже, ми обрали які чинники будуть впливають на нашу систему під час її роботи та визначили результат її роботи.

2.2 Дерево функцій системи

На даному етапі нам необхідно визначити головні та побічні функції системи. Це потрібно для того щоб зрозуміти які кроки повина виконати система для успішного функіонування. Потрібно побудувати послідовність виконання функцій нашої системи.

Дерево функцій (Function Tree) - ієрархічна модель видів діяльності підприємства, що забезпечують досягнення дерева цілей.

Вершиною дерева функцій є головна мета підприємства, гілки дерева являють собою функції (або роботи), які необхідно реалізувати для досягнення головної мети підприємства і підлеглих їй цілей нижнього рівня.

На рис 2.2 зображено дерево функцій нашої моделі.

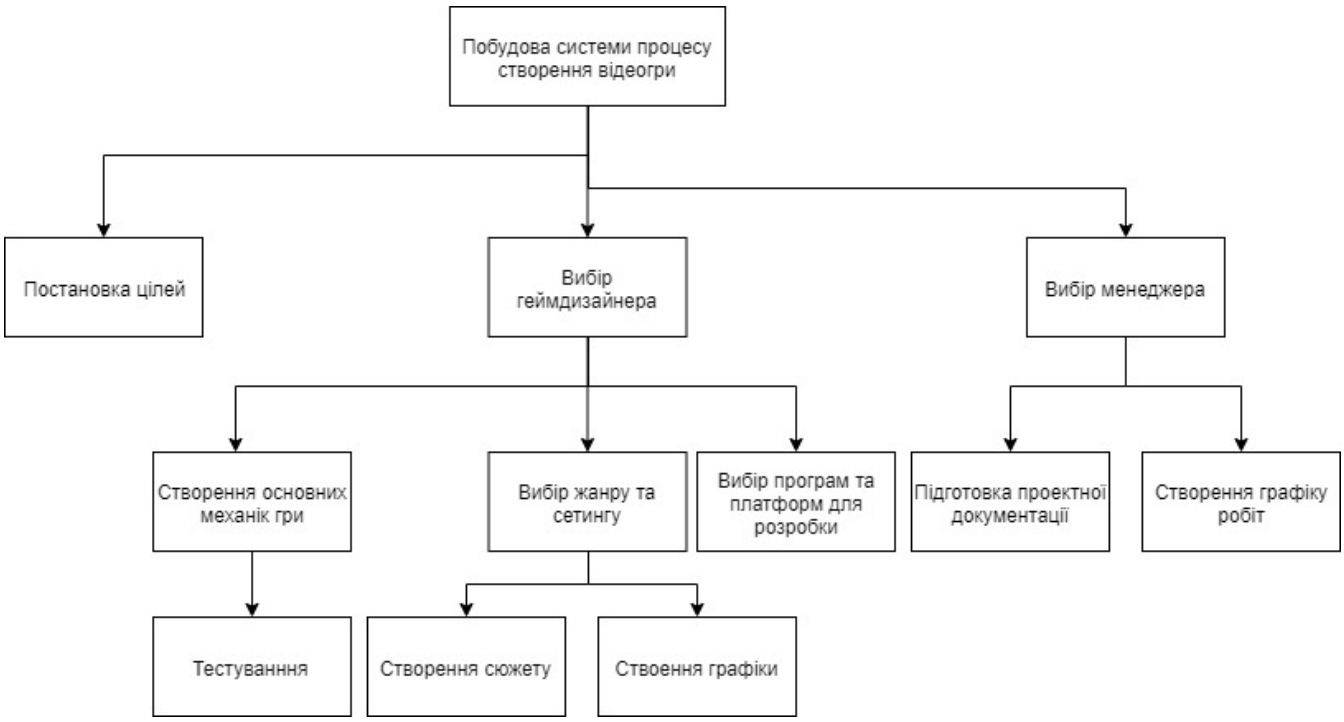

Рисунок 2.2 Дерево функцій системи

Ієрархія дерева функцій

1. Постановка цілей
2. Вибір гейм-дизайнера

2.1 Вибір програм та платформ для розробки

2.2 Вибір жанру та сетингу

2.2.1 Створення сюжету

2.2.2 Створення графіки

2.3 Створення основних ігрових механік

2.3.1 Тестування

3. Вибір менеджера

3.1 Підготовка проектної документації

3.2 Створення графіку робіт

2.3 Діаграма прецедентів.

Після визначення цілей та функцій системи необхідно вказати хто саме відповідатиме за їх виконання.

Суть даної діаграми полягає в наступному: проектована система представляється у вигляді безлічі сутностей чи акторів, що взаємодіють із системою за допомогою так званих варіантів використання. Варіант використання (англ. use case) використовують для описання послуг, які система надає актору. Іншими словами, кожен варіант використання визначає деякий набір дій, який виконує система при діалозі з актором. При цьому нічого не говориться про те, яким чином буде реалізована взаємодія акторів із системою.

Дана діаграма прецедентів містить таких основних акторів як гейм-дизайнер, менеджер та замовник та їхні прецеденти, а саме: Вибір програм для розробки, Вибір жанру та сетингу, Створення основних ігрових механік, Підготовка проектної документації, Створення робочого плану, Створення систем монетизації та реклами, Постановка цілей, Фінансування, Випуск гри.

Спочатку Замовник виконує такі дії як постановка цілей та фінансування на базі цих дій геймдизайнер виконує: вибір програм для розробки, вибір жанру та сетингу та створює основні ігрові механіки. На основі фінансування та роботі геймдизайнера менеджер повинен створити робочий план, підготувати проектну документацію, а також створити системи монетизації та реклами гри. Після створення гри замовнику потрібно випустити гру.

Рисунок 2.3 Діаграма основних прецедентів системи

Дана діаграма прецедентів (рис. 2.3) містить таких акторів як тестувальники, сценаристи, дизайнери, розробники це саме ті люди які створюють гру. Та їхні прецеденти, а саме: Тестування продукту, написання відгуку, Створення сюжету, Написання діалогів та інших текстових елементів гри, Створення концепт-артів та загального візуального стилю гри, створення графічної складової, Написання програмного коду, впровадження ігрових механік, Створення файлової архітектури проекту.

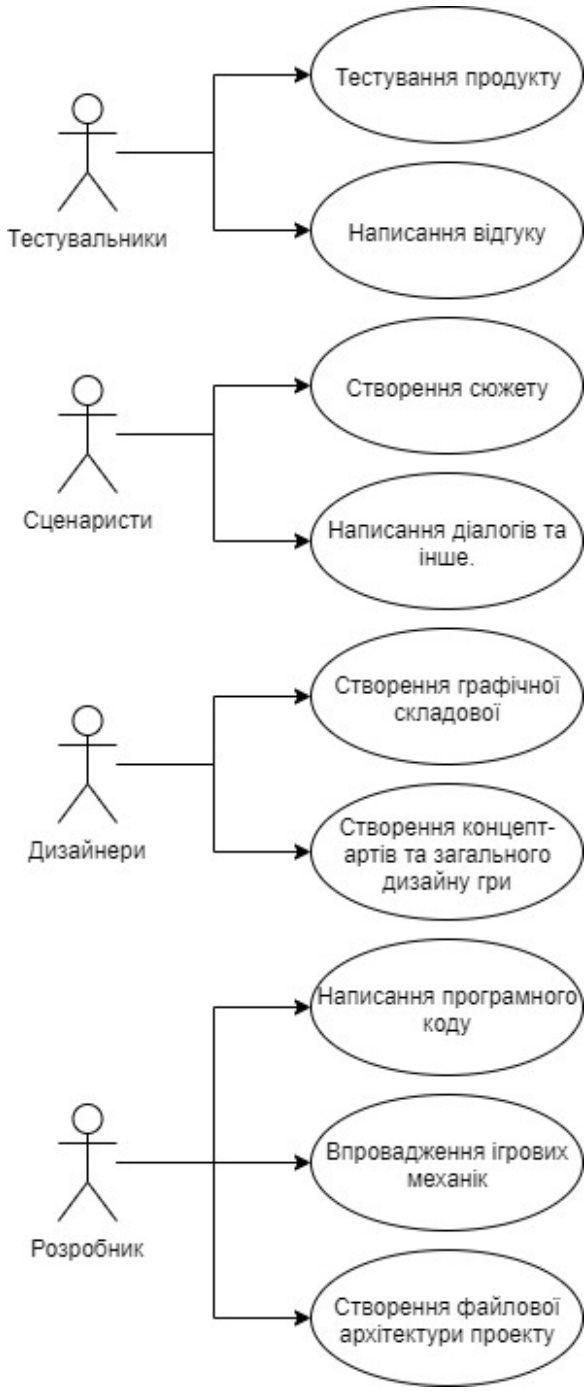

Рисунок 2.4 Діаграма побічних прецедентів системи

Після виконаних таких дій менеджера як створення робочого плану дані актори починають працювати. Дизайнери, сценаристи та розробники на базі рішень геймдизайнера починають виконувати свої дії (геймдизайнер продовжує керувати та приймати рішення на протязі всього життєвого циклу проекту). А після виконання своїх дій надсилають результат тестувальникам які тестують продукт та дають відгук на базі якого проект змінюється чи дороблюється. Після чого кінцевий результат відправляється замовнику.

2.4 Діаграма класів

Діаграма класів – це статичне представлення структури моделі. Відображає статичні (декларативні) елементи, такі як: класи, типи даних, їх зміст та відношення. Діаграма класів може містити позначення для пакетів та може містити позначення для вкладених пакетів. Також, діаграма класів може містити позначення деяких елементів поведінки, однак їх динаміка розкривається в інших типах діаграм. Діаграма класів служить для представлення статичної структури моделі системи в термінології класів об'єктно-орієнтованого програмування. На цій діаграмі показують класи, інтерфейси, об'єкти й кооперації, а також їхні відносини.

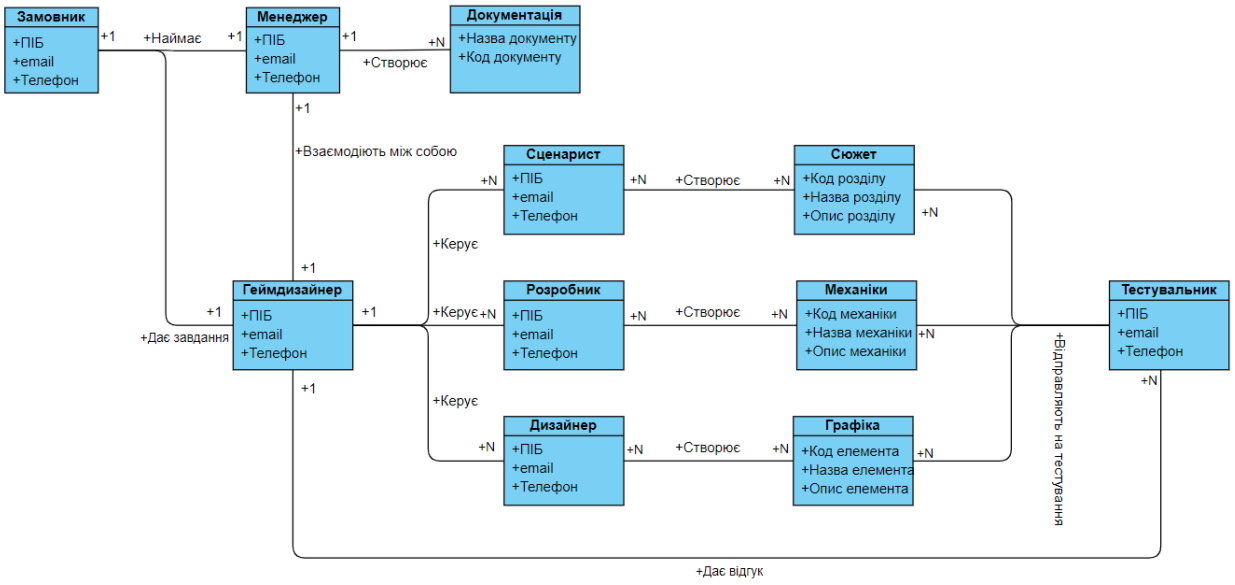

Рисунок 2.5 Діаграма класів системи

Дана діаграма містить такі класи і зв’язки між ними:

- Замовник та зв’язані з ним класи Менеджер(1…1), Геймдизайнер(1…1).
- Менеджер зв’язаний з класами Документація(1…N) яку він створює, Геймдизайнер(1…1) з яким він взаємодіє.
- Геймдизайнер пов’язаний з класами Сценарист(1…N), Розробник(1…N), Дизайнер(1…N) якими він керує.
- Сценарист зв’язаний з класом Сюжет(N…N).

- Розробник зв’язаний з класом Механіки(N…N).
- Дизайнер зв’язаний з класом Графіка(N…N).
- Тестувальник пов’язаний з класами Сюжет(N…N), Механіки(N…N), Графіка(N…N), Геймдизайнер(N…1).

2.5 Діаграма послідовності

Діаграма послідовності відображає взаємодії об'єктів впорядкованих за часом. Зокрема, такі діаграми відображають задіяні об'єкти та послідовність відправлених повідомлень.

На діаграмі послідовностей показано у вигляді вертикальних ліній різні процеси або об'єкти, що існують водночас. Надіслані повідомлення зображуються у вигляді горизонтальних ліній, в порядку відправлення.

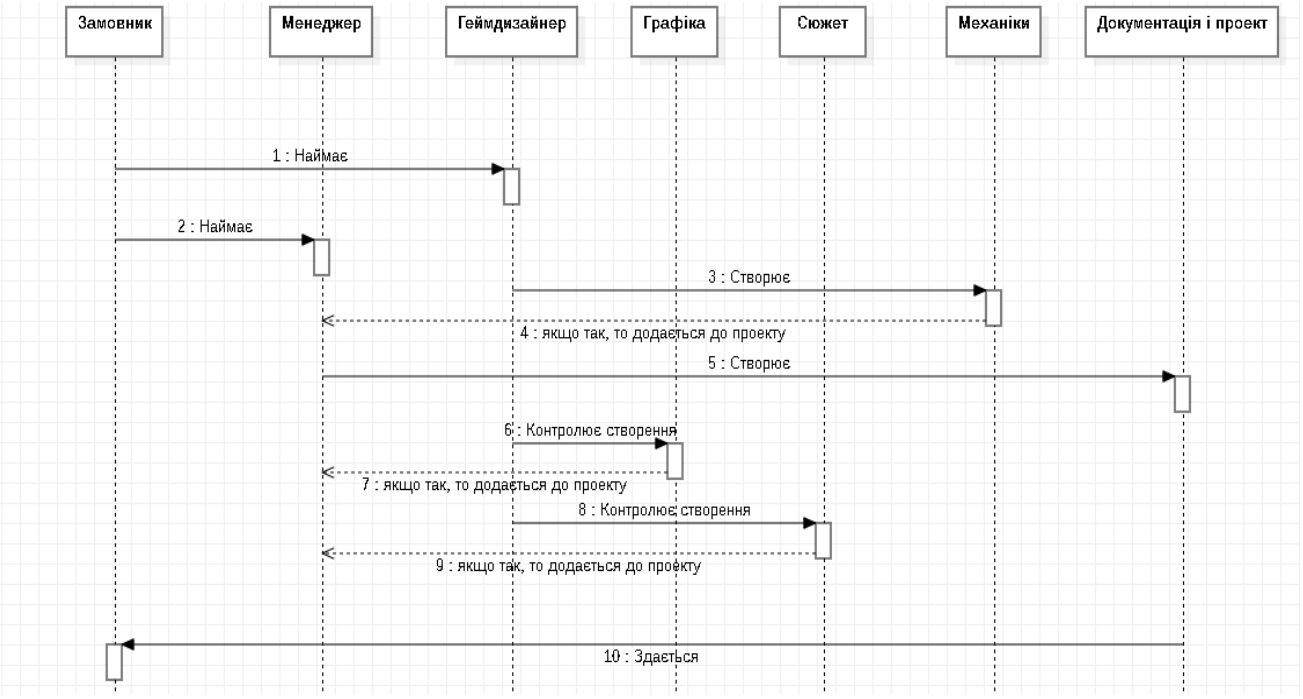

Рисунок 2.6 Діаграма послідовності

Дана діаграма послідовності описує роботу системи в цілому. Процес відбувається таким чином:

1. Замовник наймає менеджера.

2. Замовник наймає геймдизайнера.
3. Геймдизайнер створює механіки та саму концепцію відеогри.
4. Геймдизайнер контролює створення графіки дизайнерами.
5. Геймдизайнер контролює створення сюжету сценаристами.
6. Графіка додається до проекту після проходження перевірки.
7. Сюжет додається до проекту після проходження перевірки.
8. Механіка додається до проекту після проходження перевірки.
9. Менеджер створює проекту документацію та оформлює проект.
10. Готовий проект здається замовнику.

2.6 Діаграма діяльності

Діаграма діяльності - це технологія, що дозволяє описувати логіку процедур, бізнес-процеси і потоки робіт. У багатьох випадках вони нагадують блок-схеми, але принципова різниця між діаграмами діяльності і нотацією блок-схем полягає в тому, що перші підтримують паралельні процеси.

Саме діаграма діяльності допоможе описати логіку поведінки системи. Можна побудувати кілька діаграм діяльності для однієї і тієї ж системи, причому кожна з них буде фокусуватися на різних аспектах системи, показувати різні дії, що виконуються всередині неї.

Саме на діаграмі діяльності представлені переходи від однієї діяльності до іншої. Це, по суті, різновид діаграми станів, де всі або більша частина станів є деякими функціями, а все або більшість переходів спрацьовують при завершенні певної діяльності і дозволяють перейти до виконання наступної.

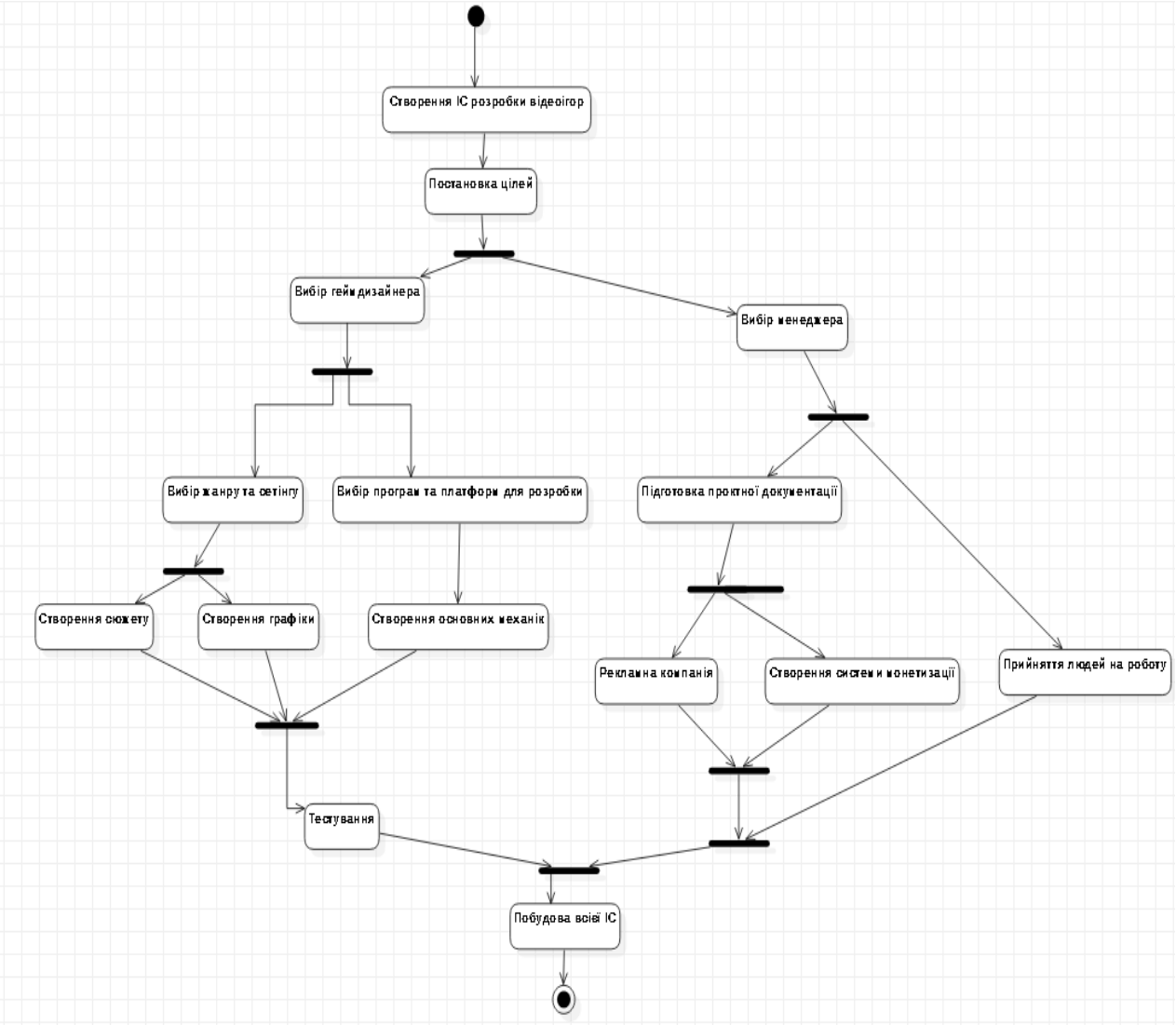

Рисунок 2.7 Діаграма діяльності

На моїй діаграмі (рис. 2.7) можна побачити 6 основних елементів таких як:

- Початок всієї діаграми;
- Кінець діаграми;
- Action1 - це ActionBox у нього записуються усі наші дії;

Для правильного розташування наших ActionBox, нам потрібні 3 елементи, а саме:

- Fork – це елемент, який дозволяє паралельно проводити ActionBox;
- Join – це елемент, який об’єднує наші паралельні активності;
- Control Flow – це елемент, який поєднує наші бокси.

3 РОЗРОБКА ВІДЕОГРИ НА БАЗІ ІНФОРМАЦІЙНОЇ СИСТЕМИ

3.1 Опис особливостей реалізованого проекту

Метою проекту відеогри є створення проекту-головоломки з використанням механік двовимірного платформера на основі створеної системи розробки відеогри. Головною особливістю гри є використання двох ігрових персонажів у правильній послідовності для завершення рівня.

Гравець має перемикати контроль між роботами, які мають різний набір можливостей, для досягнення мети. Сильний робот на гусеницях може штовхати ящики та підіймати важкі речі, а потужні магніти в його руках можуть зрушити металеві речі для ходу назад, та він дуже повільний. Йому допомагає спритний робот на сильному моноколесі, що має більшу швидкість, може стрибати, підбирати невеликі предмети та активувати механізми, але його мобільність може сильно зменшити перша ж перешкода, яка не становить ніякої загрози для першого робота.

Тож, контролюючи двох різних персонажів, гравець має прикривати слабкі сторони кожного з роботів та застосовувати переваги конструкції кожного з них для того, щоб зібрати шестерні на рівні, для того, щоб вони мали змогу відремонтувати пускову установку, після чого рівень вважається пройденим.

Головним завданням гравця є виконання дій у певному порядку, що приведе його до закінчення рівня. Наприклад, скинути ящика за допомогою сильного робота – Гусінь, що швидкий – Колесник – міг проїхати, чи натиснути за робота з колесом кнопку, щоб у Гусіні з’явилися нові речі, з якими він може взаємодіяти, просуваючи рівень далі.

Стартовий рівень пропонує гравцю невелику головоломку з обмеженим часом – перемикаючи роботів та знаючи план виконання дій, він повинен за визначений час зібрати потрібні предмети.

Єдиною загрозою для життя роботів є провал в лабораторії, в якій вони знаходяться – якщо хтось з них впаде, рівень почнеться спочатку.

Тож, гра надає невелику загадку з таймером, що використовує механіку особливостей двох ігрових персонажів для обмеження шляхів виконання завдання, тим самим створюючи важливість виконання певних дій у певному порядку, завдяки чому гравець зможе завершити рівень.

3.2 UML - діаграма прецедентів компонентів відеогри

Необхідно описати як компоненти відеогри будуть виконувати свої функції та взаємодіяти між собою.

UML - уніфікована мова моделювання (Unified Modeling Language) - це система позначень, яку можна застосовувати для об'єктно-орієнтованого аналізу і проектування. Його можна використовувати для візуалізації, специфікації, конструювання та документування програмних систем.

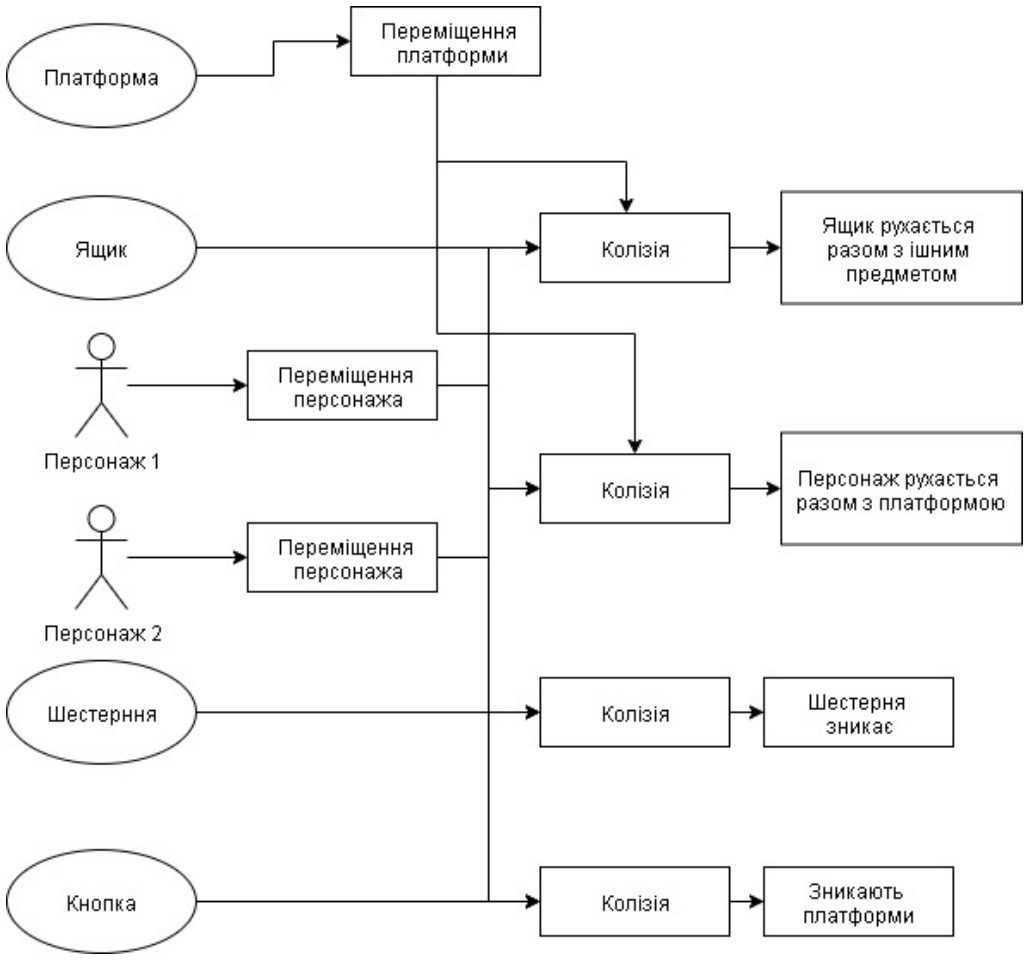

Рисунок 3.1 Діаграма прецедентів компонентів відеогри.

Опишемо основні прецедентів:

1. Переміщення персонажів та платформи

Зацікавленість та вимоги. Персонаж 1 може переміщатися по горизонталі самостійно. Персонаж 2 може переміщатися по горизонталі та вертекалі самостійно. Платформа рухається вниз та вгору почергово та постійно.

Передумови. Гра була розпочата.

Результат. Платформа може створювати колізії з персонажами, та переміщувати їх.

2. Перемога

Зацікавленість та умови. Персонаж 2 зібрав всі шестерні.

Передумова. Кількість зібраних шестернь рівна 2.

Результат. Гра починається знову.

3. Поразка

   Зацікавленість та умови. Персонаж 2 та персонаж 1 випав за межі гри

   Передумова. Персонаж нижче останньої платформи. Закінчився час.

   Результат. Гра починається знову.

4. Колізія

   Зацікавленість та умови. Персонаж 1, персонаж 2, шестерння, кнопка, ящик. Зіткнення.

   Передумова. Перетин колайдерів.

   Результат. Персонаж 1 рухає ящик. Персонаж натискає кнопку та підбирає шестерню

3.3 UML- діаграма класів коду проекту

Діаграма класів використовуються при моделюванні ПС найбільш часто. Вони є однією з форм статичного опису системи з точки зору її проектування. Діаграма класів не відображує динамічну поведінку об'єктів зображених на ній класів. На діаграмах класів показуються класи, інтерфейси і відносини між ними.

Діаграма класів - це відмінний спосіб візуалізувати класи у вашій системі, перш ніж ви почнете їх кодувати. Вони являють собою зріз структури вашої системи.

Саме діаграма класів дає нам найбільш повне і розгорнуте уявлення про структуру та зв'язки в програмному коді. Розуміння принципів побудови даної діаграми дозволяє коротко і прозоро висловлювати свої думки та ідеї.

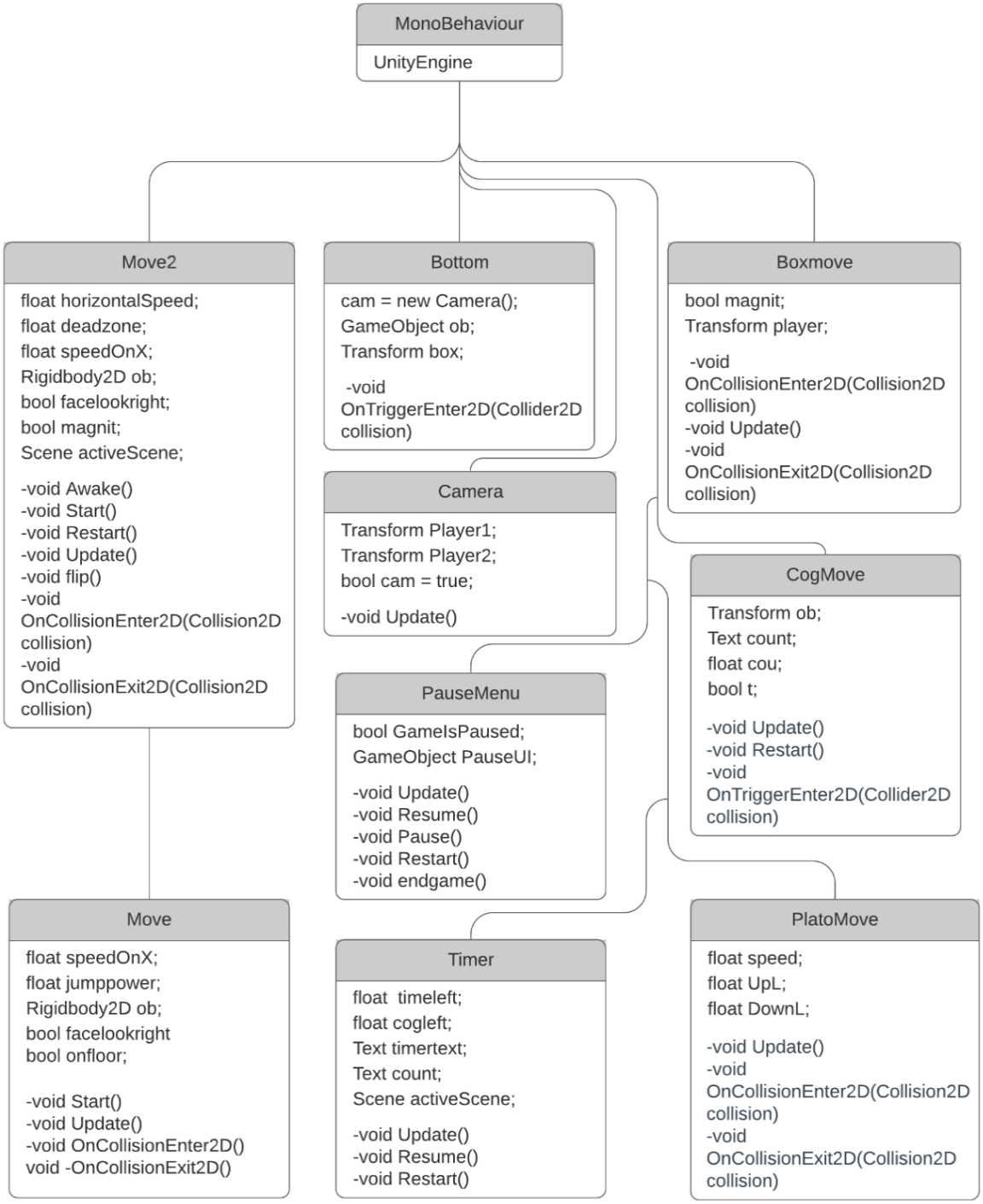

Рисунок 3.2 Діаграма класів проекту.

Функції які виконують класи:

- Клас Move2 відповідає за управління першим персонажем.
- Клас Move відповідає за управління другим персонажем.
- Клас Timer відповідає за обмеження для гри – час для виконання та умову закінчення рівня.

- Клас PlatoMove відповідає за поведінку рухомих платформ.
- Клас CogMove відповідає за поведінку шестерень.
- Клас PauseMenu відповідає за організацію та функції меню паузи.
- Клас BoxMove відповідає за правильну роботу об’єктів, що можна штовхати – ящиків.
- Клас Camera відповідає за поведінку камери гравця, тобто за те, що саме побачить гравець.
- Клас Bottom відповідає за поведінку натискання кнопки.

3.4 Опис тестових прикладів виконання програми

1) Рівень починається з того, що Колесник знаходиться посеред сцени. Йому необхідно перейти до рухомої платформи, що знаходиться зліва від гравця.

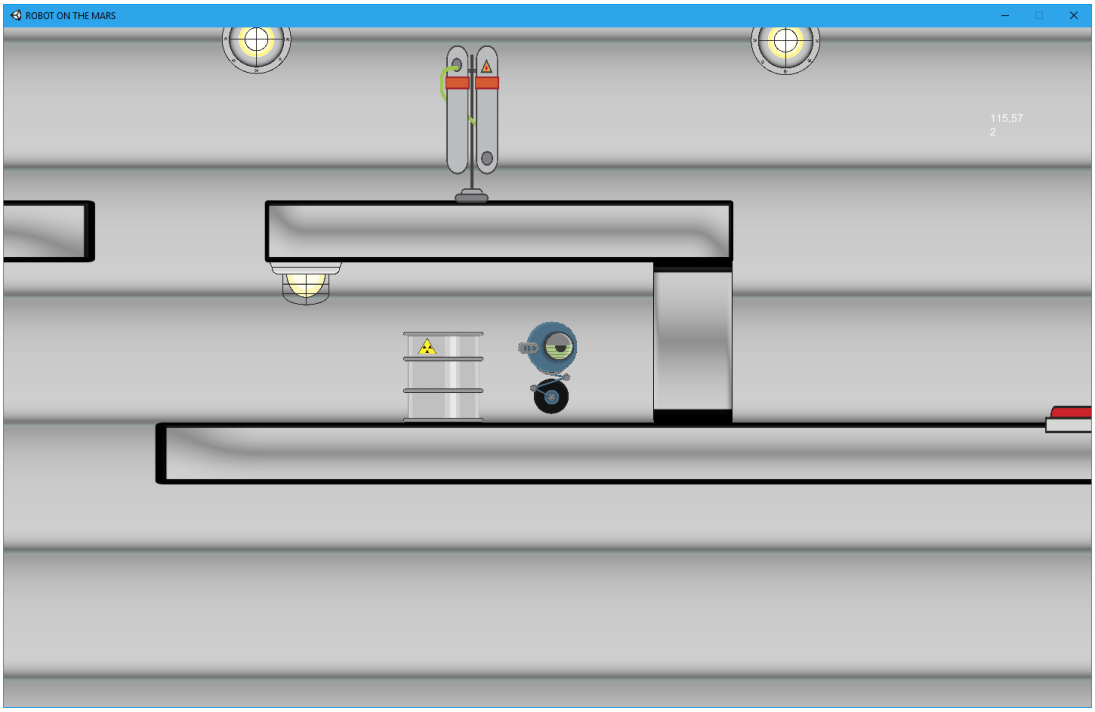

Рисунок 3.3 Приклад виконання програми

2) Після того, як Колесник підійшов до платформи, він не зможе за стрибнути на неї через ящика. Саме тому потребується допомога Гусіні. Перемикаємо контроль з Колесника на іншого робота.

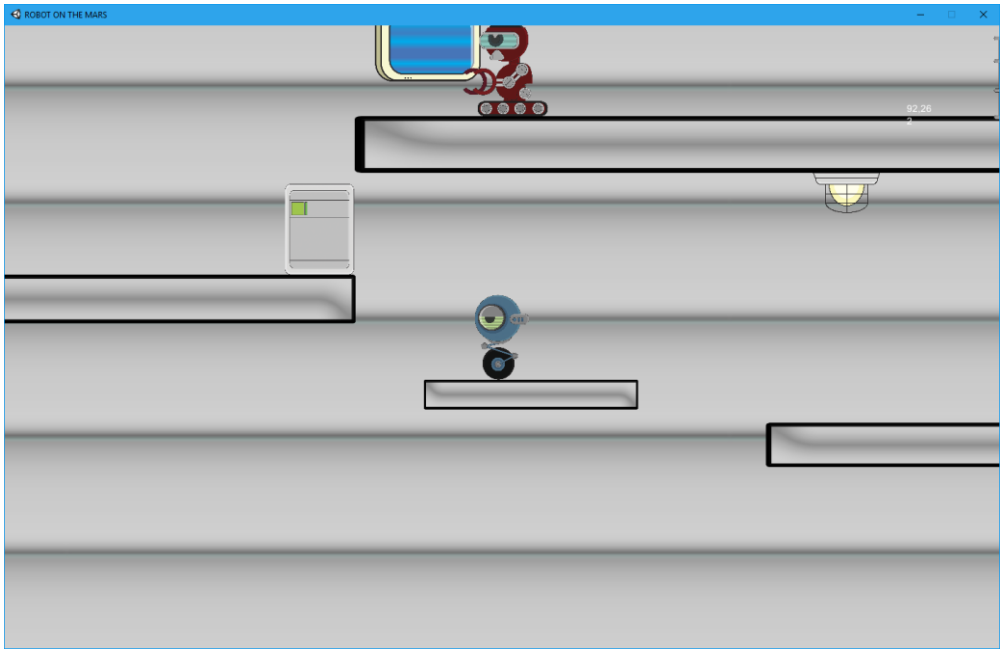

Рисунок 3.4 Приклад виконання програми

3) Через свою силу, Гусінь може зсунути ящик з місця та звільнити прохід Колеснику. Та гравець має бути обачним – потрібно здобути шестерню, яка знаходиться зліва від рухомої платформи.

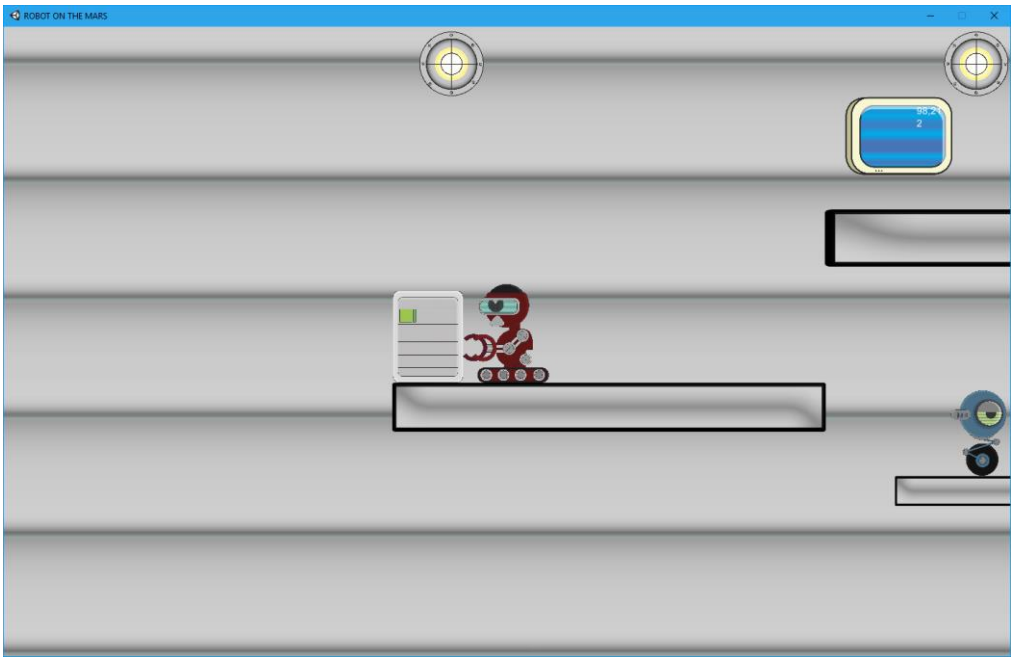

Рисунок 3.5 Приклад виконання програми

4) Після того, як Гусінь доведе ящик до краю платформи, Колесник має за стрибнути на нього. Цього вистачить, щоб з такої висоти, під час того, як елеватор дійде до своєї максимально високої точки, дострибнути до платформи зліва та забрати шестерню.

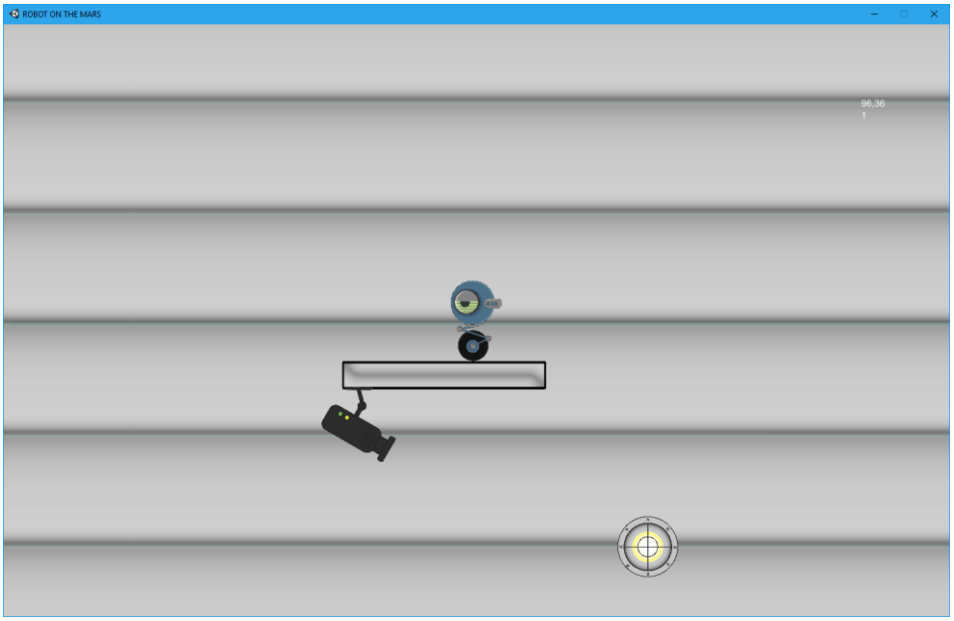

Рисунок 3.6 Приклад виконання програми

5) Доступу до другої шестерні поки що немає – для цього потрібно скинути ящики, на одному з яких вона і знаходиться. Для цього треба отримати доступ до кнопки, що вимикає живлення платформ, на яких вони тримаються.

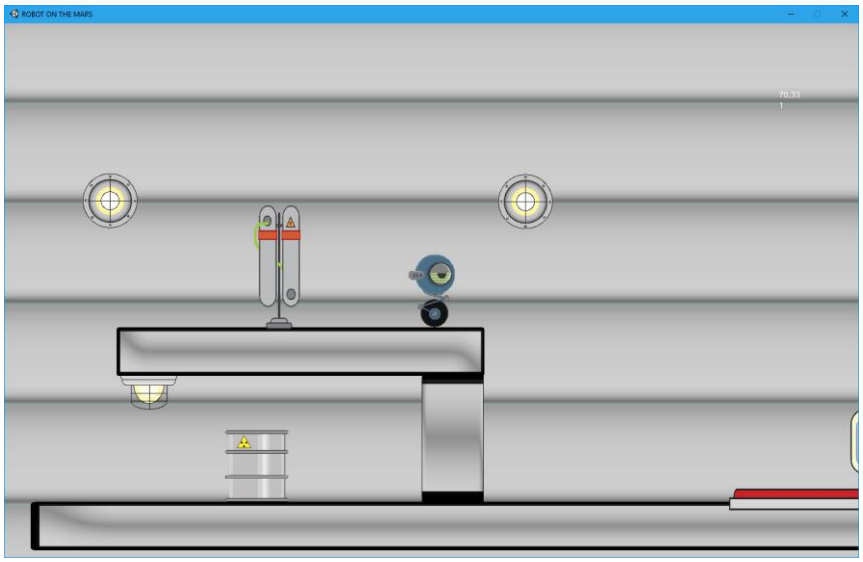

Рисунок 3.7 Приклад виконання програми

6) Колесник активує кнопку, після два ящики падають. Саме за допомогою них Гусінь повинен забезпечити собі проїзд до Колесника та скинути йому ящика для того, щоб його стрибка вистачило для того, щоб вибратися з низької платформи.

Рисунок 3.8 Приклад виконання програми

7) Гравець повинен перевести контроль на Гусінь, після чого почати скидати ящики, що розділяють його та Колесника. Під час цього процесу він знайде останню шестерню, які, на жаль, може підбирати лише другий робот.

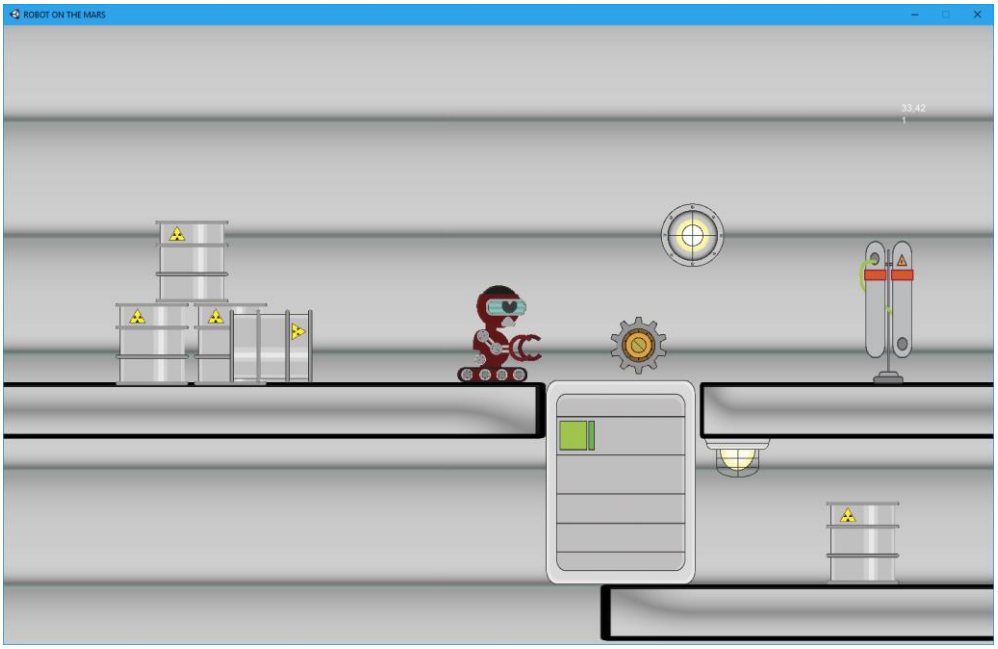

Рисунок 3.9 Приклад виконання програми

8)Після того, як другий ящик був скинутий, Колесник матиме змогу вибратися з ями та за стрибнути на платформу вище. Після цього, він підбирає останню шестерню та рівень вважається завершеним.

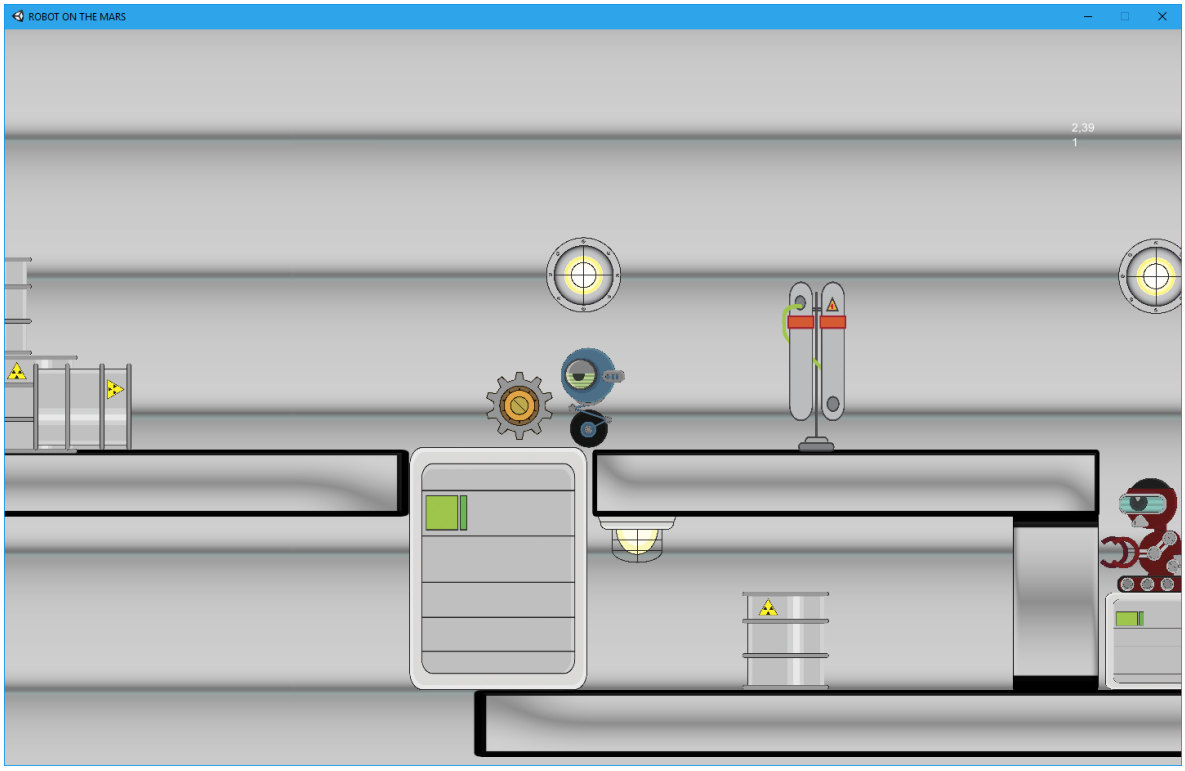

Рисунок 3.10 Приклад виконання програми

4 СИСТЕМА ОЦІНКИ ЯКОСТІ ВІДЕОГРИ ЗА КРИТЕРІЯМИ

4.1 Аналіз ситуаційних умов прийняття рішень та побудова сценарію роботи системи

Зміст. Ставиться задача провести оцінку заданих відеоігор в установленій шкалі з метою визначення найкращої серед них на основі врахування ряду критеріїв та визначення найголовніших критеріїв при розробці відеогри.

Вхідні дані: множина відеоігор, перелік, опис критеріїв та їх вага, значення критеріїв у встановлених одиницях вимірювання, діапазон допустимих значень критеріїв («ідеальне» значення – найгірше допустиме значення) у встановлених одиницях вимірювання.

Вихідні дані: комплексна оцінка кожного варіанта рішень, рейтинг відеоігор, найкраща гра, найважливіші критерії.

Кроки сценарію:

1. Приведення значень критеріїв до єдиної шкали вимірювання на основі метода лінійної нормалізації у встановленому діапазоні значень.
2. Розрахунок комплексної оцінки кожної відеогри на основі метода лінійної згортки критеріїв.
3. Перевірка погодженості думок експертів.
4. Встановлення рейтингу відеоігор.

Визначення найкращої відеогри ОПР (особою, що приймає рішення).

Рисунок 4.1 Модель чорної скриньки системи

4.2 Структурно-функціональний аналіз

Структурно-функціональний аналіз — метод системного дослідження соціальних явищ і процесів як структурно розчленованої цілісності, де кожний елемент структури має певне функціональне призначення.

Передбачає два основні підходи:

1) структурний, за якого науковий пошук починається з аналізу структур, а кінцевою метою є виявлення функцій, що ними виконуються;

Дерево функцій (Function Tree) - ієрархічна модель видів діяльності підприємства, що забезпечують досягнення дерева цілей.

Вершиною дерева функцій є головна мета підприємства, гілки дерева являють собою функції (або роботи), які необхідно реалізувати для досягнення головної мети підприємства і підлеглих їй цілей нижнього рівня.

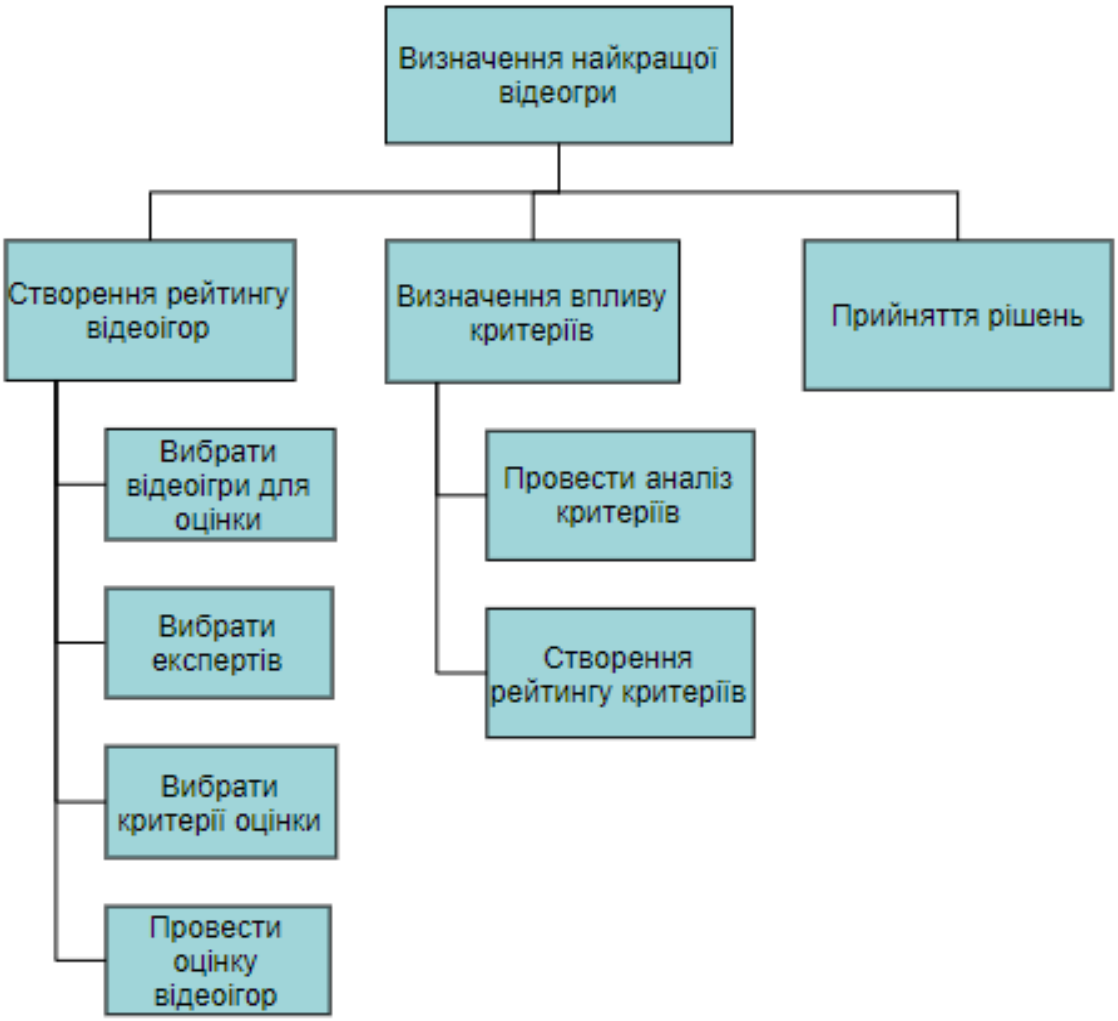

Рисунок 4.2 Дерево функцій

2) функціональний — постулюється певна сукупність функціональних вимог і потім виявляються різні структури, що здійснюють ці функції.

Діаграма прецедентів — на якій зображено відношення між акторами та прецедентами в системі. Також, перекладається як діаграма варіантів використання.

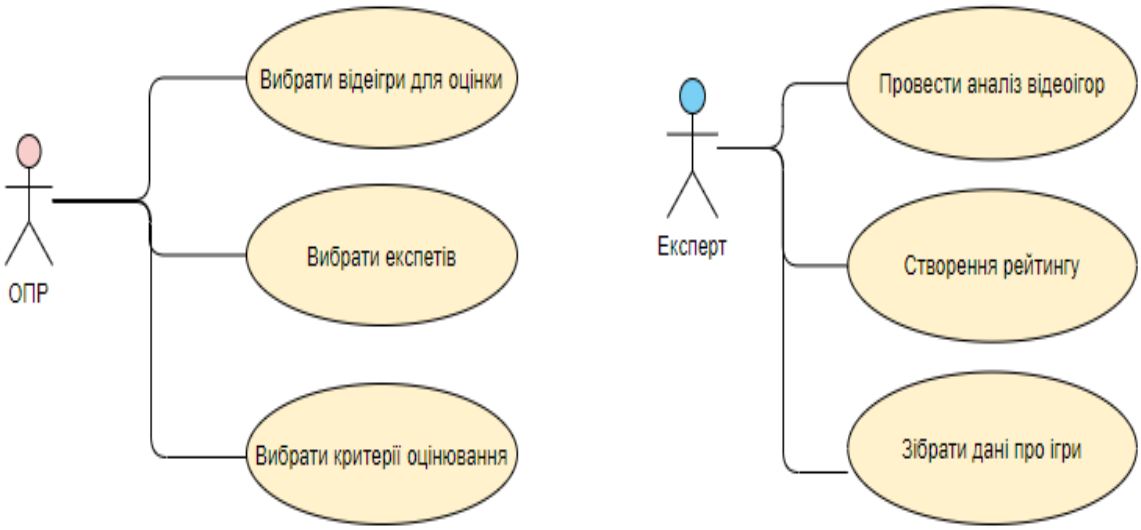

Рисунок 4.3 Дерево прецедентів

Суть даної діаграми (рис. 4.3) полягає в наступному: проектована система представляється у вигляді безлічі сутностей чи акторів, що взаємодіють із системою за допомогою так званих варіантів використання. Варіант використання (англ. use case) використовують для описання послуг, які система надає актору. Іншими словами, кожен варіант використання визначає деякий набір дій, який виконує система при діалозі з актором.

4.3 Статичні та динамічні моделі опису архітектури системи та технології її роботи

Діагра́ма кла́сів — статичне представлення структури моделі. Відображає статичні (декларативні) елементи, такі як: класи, типи даних, їх зміст та відношення. Діаграма класів може містити позначення для пакетів та може містити позначення для вкладених пакетів. Також, діаграма класів може містити позначення деяких елементів поведінки, однак їх динаміка розкривається в інших типах діаграм. Діаграма класів служить для представлення статичної структури моделі системи в термінології класів

об'єктно-орієнтованого програмування. На цій діаграмі (рис 4.4) показують класи, інтерфейси, об'єкти й кооперації, а також їхні відносини.

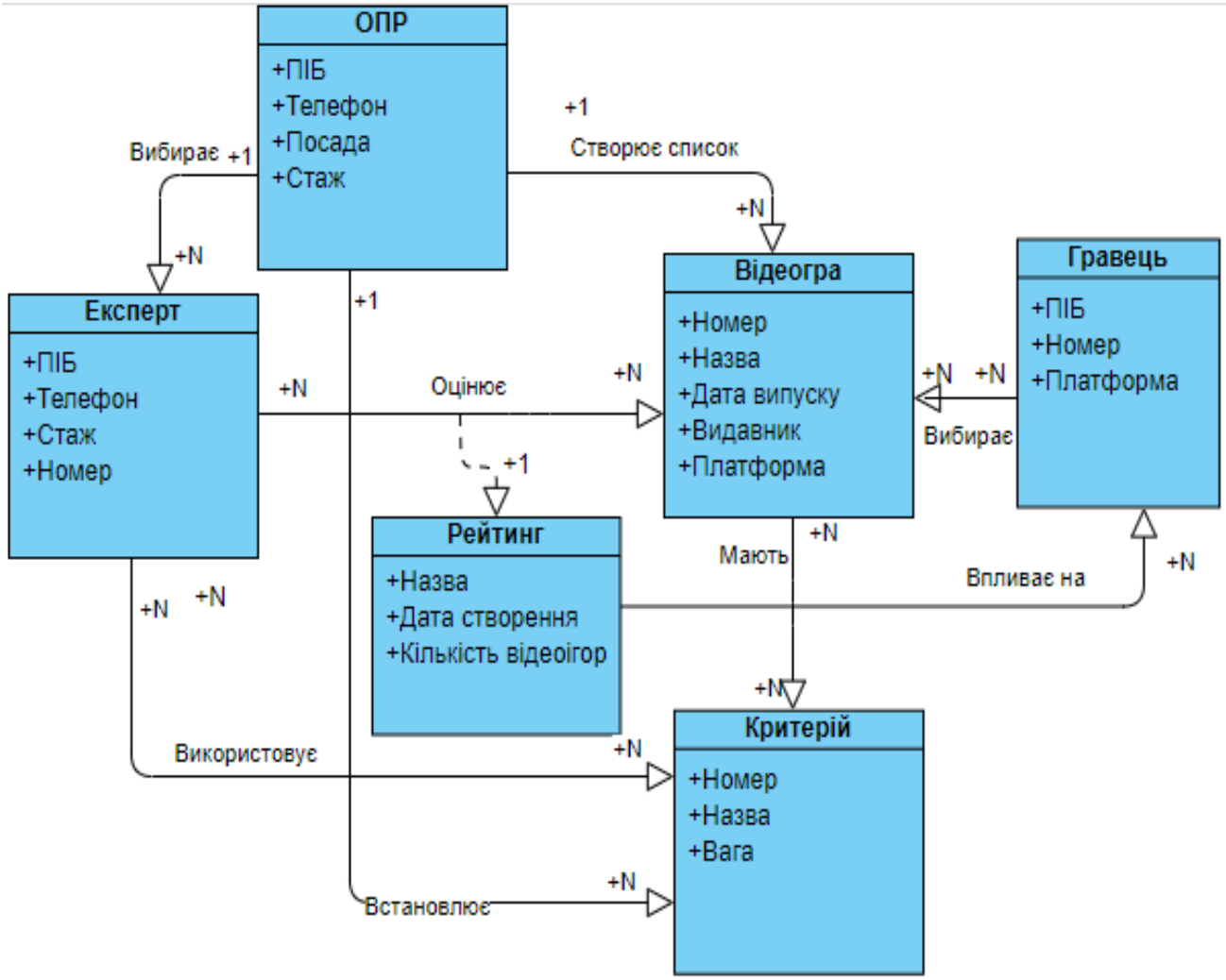

Рисунок 4.4 Діаграма класів

Головним класом являється ОПР. Він має такі напрямлені асоціативні зв’язки з іншими класами, такими як експерт, відеогра, критерії.

Діаграма послідовності відображає взаємодії об'єктів впорядкованих за часом. Зокрема, такі діаграми відображають задіяні об'єкти та послідовність відправлених повідомлень.

На діаграмі послідовностей показано у вигляді вертикальних ліній різні процеси або об'єкти, що існують водночас. Надіслані повідомлення зображуються у вигляді горизонтальних ліній, в порядку відправлення.

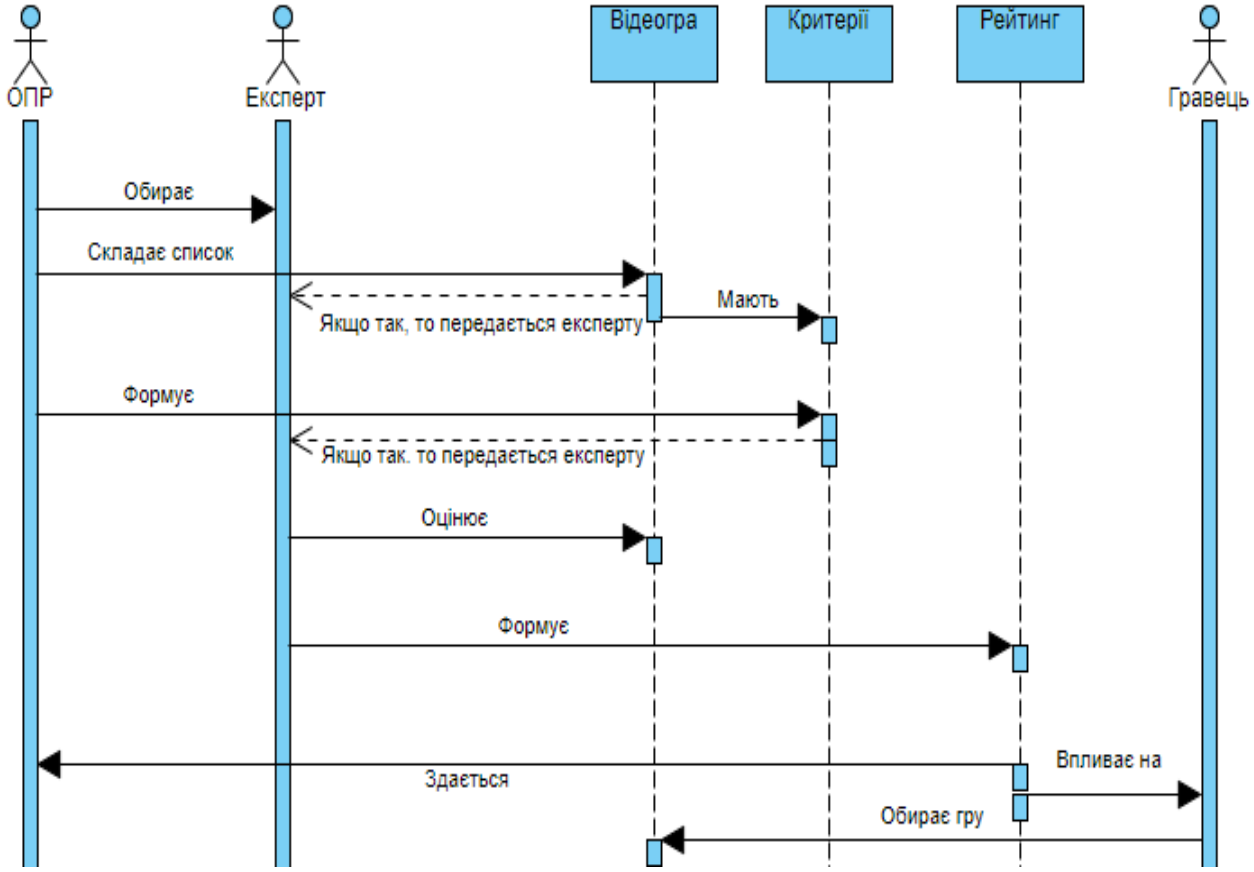

Рисунок 4.5 Діаграма послідовностей

На даній діаграмі (рис. 4.1) зображено послідовна побудова «ОПР».

Далі поступово відбуваються зв’язки з іншими боксами такими як:

- Експерт;
- Відеогра;
- Критерії;
- Рейтинг;
- Гравець;

А також під час побудови цієї діаграми використовувались два типи зв’язків, а саме «Message» та «Reply Message».

4.4 Метод розв’язання визначеної задачі (функціонального модуля системи) та математична постановка її реалізації

4.4.1 Метод розв’язання визначеної задачі

Якщо ОПР визначає різну значущість критеріїв і вона однакова на всьому діапазоні їх зміни, один з найбільш розповсюджених методів, що може бути запропонований – метод, що базується на принципі лінійної згортки критеріїв

В даному методі важливою умовою його ефективності є адекватне визначення «удільної ваги» критеріїв. Найбільш розповсюджені шляхи їх визначення – розрахунковий або на основі експертних оцінок.

Розрахунковий шлях базується на можливості умовного приведення оціночних показників варіантів до загальної одиниці вимірювання ефективності рішень.

Методи визначення ваги критеріїв експертним шляхом різноманітні. Врахування погляду різних фахівців позитивно впливає на адекватність оцінок. Шлях експертного визначення значущості критеріїв при якісній організації цієї роботи дозволяє враховувати при виборі варіантів різні не зовсім формалізує фактори, що можуть суттєво впливати на пріоритет критеріїв. При цьому важливо, щоб в оцінці порівняльної важливості критеріїв приймали участь особи, що компетентні в визначенні пріоритетів прийняття рішень.

Рангом називається ступінь відмінності по якійсь ознаці, а ранжуванням — процес визначення рангів, відносних кількісних оцінок ступенів відмінностей за якісними ознаками (наприклад, розташування факторів у порядку їх суттєвості, значимості в даному дослідницькому контексті). Ранжування застосовується у випадках, коли неможлива або недоцільна безпосередня оцінка. При цьому "ранжування об'єктів містить

лише інформацію про те, який з об'єктів кращий, і не містить інформацію про те, наскільки або у скільки разів один об'єкт переважає інший.

Метод полягає у віднесенні оціненого об'єкта до певного значення за оціночною шкалою (в присвоєнні об'єкту оцінки балів в певному інтервалі). Наприклад, оцінка від 0 до 10 згідно із перевагою по якійсь ознаці або їх групі (наприклад, альтернативі — за перевагою, критерію — за значущістю, фактором зовнішнього середовища — по впливу, проблемі — за пріоритетністю рішення).

Для подальшої обробки отримані оцінки можуть бути пронормовані, тобто їх сума може бути приведена до одиниці шляхом ділення кожної оцінки на їх загальну суму. Для наочності і зручності обробки (здійснення вибору ОПР) оцінки можуть бути переведені в ранги. Максимальній оцінці при цьому відповідає найвищий ранг, тобто 1, а мінімальний — n(при n — кількості оцінених об'єктів).

При оцінці об'єкта за кількома параметрами (наприклад, при оцінці альтернатив за кількома критеріями) сумарна оцінка об'єкта проводиться наступним чином.

Експерти виносять судження про вагу параметрів (наприклад, про ваги критеріїв) і оцінки об'єкта згідно всіх параметрів (наприклад, оцінки альтернатив за критеріями).

Аналітики обробляють отримані оцінки. Обчислюють нормалізовану вагу параметрів (наприклад, критеріїв) за формулами середнього арифметичного, середнього геометричного або середньозваженого.

Потім комплексні оцінки нормалізують. І роблять висновки на базі отриманих результатів.

4.4.2 Математична постановка задачі

Загальна характеристика задачі

1. ОПР надає критерії оцінювання («визначення критеріїв»), а також експерти задають їх вагу («задання ваги критеріїв»).
2. Відеогра проходить тестування експертами зі спеціальними критеріями («оцінювання»), результати якого подаються ОПР.
3. Результати оцінки передаються ОПР, який оцінює всі наявні дані, разом з результатами оцінювання («оцінка результатів»), формує рейтинг відеоігор («формування рейтингу»), а також обирає найкращу гру, тобто першу в рейтингу («обрання найкращої відеогри»).

4.4.3 Словник даних

Форма подання вхідної інформації

@ИМЯ = Відеогра
@ТИП = дискретний потік
@БНФ={КОД ВІДЕОГРИ + НАЗВА ВІДЕОГРИ+ ТИП ВІДЕОГРИ }

@ИМЯ = ЕКСПЕРТИ
@ТИП = дискретний потік
@БНФ={КОД ЕКСПЕРТИ+ ПІБ ЕКСПЕРТИ}

@ИМЯ = КРИТЕРІЇ
@ТИП = дискретний потік
@БНФ= {КОД КРИТЕРІЮ + ЗНАЧЕННЯ КРИТЕРІЮ + НАЗВА КРИТЕРІЮ + ВАГА КРИТЕРІЮ}

@ИМЯ = ЗНАЧЕННЯ КРИТЕРІЮ
@ТИП = дискретний потік

@БНФ= /значення критерію для визначеного працівника/

@ОДИНИЦЯ ВИМІРЮВАННЯ = бали

@ДІАПАЗОН ЗНАЧЕНЬ = 0 – 10

@ТОЧНІСТЬ ЗНАЧЕНЬ = 1

@ИМЯ = ВАГА КРИТЕРІЮ

@ТИП = дискретний потік

@БНФ= /ступінь важливості визначеного критерію/

@ОДИНИЦЯ ВИМІРЮВАННЯ = бали

@ДІАПАЗОН ЗНАЧЕНЬ = 0 – 1

@ТОЧНІСТЬ ЗНАЧЕНЬ = 1

Форма подання вихідної інформації

@ИМЯ = РЕЙТИНГ ВІДЕОІГОР

@ТИП = дискретний потік

@БНФ={КОД ВІДЕОГРИ + НАЗВА ВІДЕОГРИ + КОМПЛЕКСНА ОЦІНКА ВІДЕОГРИ}

@ИМЯ = ВАЖЛИВІСТЬ КРИТЕРІЇВ

@ТИП = дискретний потік

@БНФ= {КОД КРИТЕРІЮ + НАЗВА КРИТЕРІЮ + ВАГА КРИТЕРІЮ}

@ИМЯ = НАЙКРАЩА ВІДЕОГРА

@ТИП = дискретний потік

@БНФ= {КОД ВІДЕОГРИ + НАЗВА ВІДЕОГРИ}

@ИМЯ = КОМПЛЕКСНА ОЦІНКА ВІДЕОГРИ

@ТИП = дискретний потік

@БНФ= /сумарна оцінка для визначеного працівника/

@ОДИНИЦЯ ВИМІРЮВАННЯ = бали

@ДІАПАЗОН ЗНАЧЕНЬ = 0 – 10

@ТОЧНІСТЬ ЗНАЧЕНЬ = 1

4.5 Формалізований опис процесу по переробці вхідної інформації в вихідну

Дана діаграма діяльності на рис 4.6 демонструє процес роботи даної інформаційної системи на основі діаграми прецедентів системи.

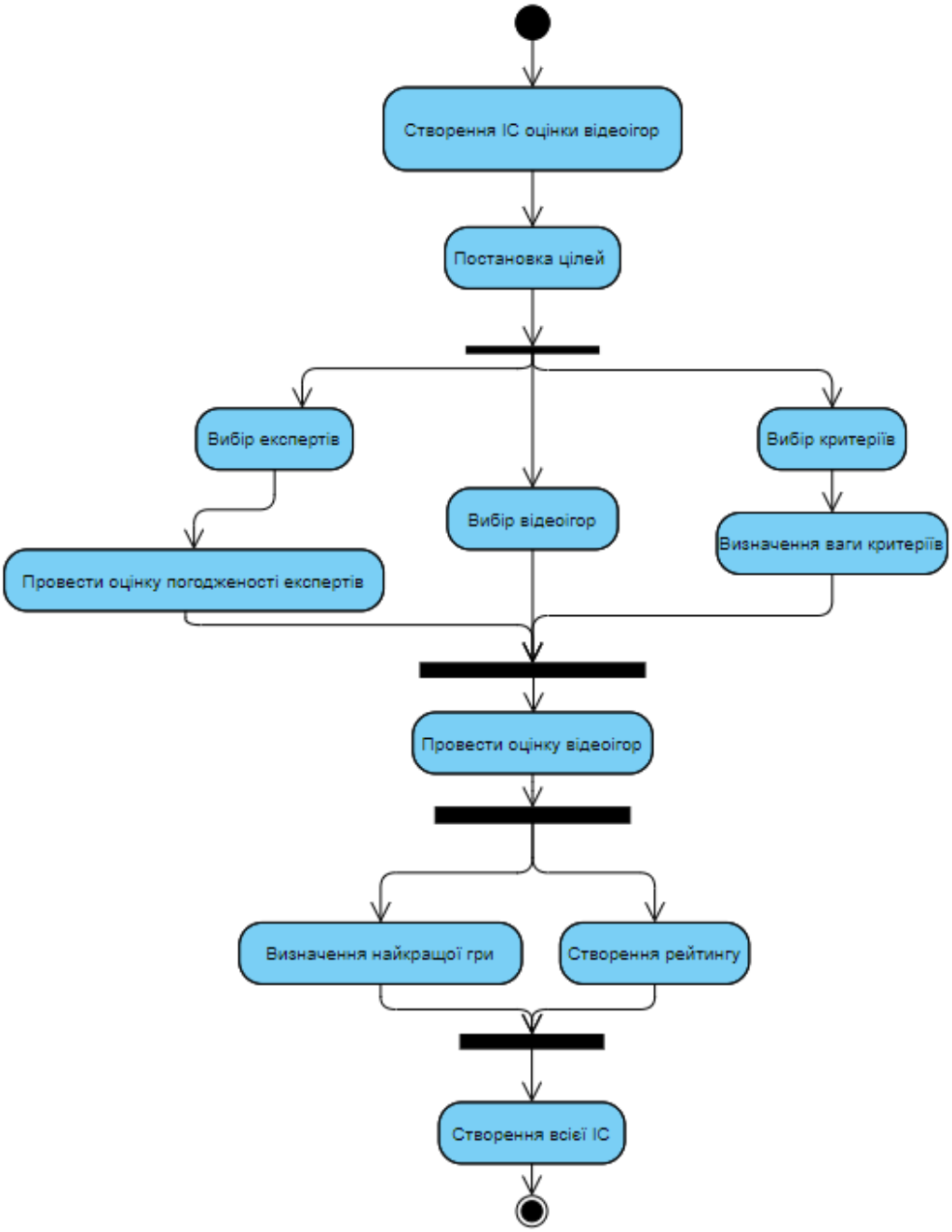

Рисунок 4.6 Діаграма діяльності

Формалізований опис процесу по переробці вхідної інформації в вихідну – опис задачі формальними засобами (символами математики і математичної логіки), що вираженні її у вигляді конкретних математичних залежностей і відношень (функцій, рівнянь, нерівностей, логічних звя’зків тощо). Спочатку зазвичай визначається основна тип математичної моделі, а потім уточнюються її деталі (конкретний перелік змінних і параметрів, форма зв’язків тощо). Представлення формалізованого опису доцільно задавати на основі логіки реалізації запропонованої схеми алгоритму або діаграми діяльності

4.6 Розрахунковий приклад

Для оцінювання 7 проектів відеоігор були вибрані наступні критерії: геймплей, графіка, сюжет, звук, оптимізація та атмосфера.

Чотири експерти залучені до оцінки ступеня впливу шести (К=6) вхідних змінних (критеріїв) на вихідну змінну об’єкта керування.

Таблиця 4.1

Таблиця критеріїв

Експерт	К1	К2	К3	К4	К5	К6
1.	3	5	1	4	2	6
2.	5	6	1	3	3	4
3.	4	5	1	3	4	5
4.	4	7	3	5	2	5

1 Нормалізація представлення матриці:

Таблиця 4.2

Нормалізована таблиця критеріїв

Експерт	К1	К2	К3	К4	К5	К6
1.	3	5	1	4	2	6

2.	5	6	1	2.5	2.5	4
3.	3.5	5.5	1	3	3.5	5.5
4.	4	7	3	4.5	2	4.5

2 Перевірка міри погодженості думок експертів:

Визначаємо середній ранг критеріїв*:*

$$Xcp = 0.5x4x7=14$$

Визначаємо дисперсію думок експертів *S* та значення $\sum_{ш=1}^{b} Ti$:

$$S= (16-14)^2+(23-14)^2+(6-14)^2+(15-14)^2+(11-14)^2+(20-14)^2=195;$$

$$\sum_{ш=1}^{b} Ti=(2^3-2)+(2^3-2)+(2^3-2)+(2^3-2)=24.$$

Визначаємо коефіцієнт конкордації думок експертів:

$$K = \frac{12x195}{4^2(6^3-6)-4x24} = 0.72$$

K=0.72, тому вважаємо думки експертів більш менш погодженими.

3 Встановити ступень впливу в шкалі 100-40.

Заданий діапазон значень від $u_{min}(40)$ до $u_{max}(100)$.При чому u_{max} присвоюється найкращому варіанту рішень по кожній властивості (варіанту з мінімальним значенням $X^l j$), u_{min} – найгіршому варіанту рішень по кожній властивості (варіанту з максимальним значенням $X^l j$).Тоді перетворений ранг варіантів може бути визначений наступним чином (4.1):

$$Uj^l = U_{\min} + \frac{x_{\max}{}^l - x^l{}_J}{X_{max}{}^l - X_{\min}{}^l}(U_{\max} - U_{\min}) \qquad (4.1)$$

Де $X_{max}{}^l$ та $X_{min}{}^l$ відповідно максимальні та мінімальні сумарні ранги, що отримані при оцінці експертами варіантів по властивості l.

Визначаємо сумарні ранги кожного критерію:

$X_j1 = 16$, $X_j2=23$, $X_j3=6$, $X_j4=15$, $X_j5=11$, $X_j6=20$.

За формулою, наданою вище, вираховуємо ранги всіх критеріїв:

$$U1 = 40 + \frac{16-6}{23-6}(100-40) = 75.2$$

U1=75.2, U2=100, U3=40, U4=71.6, U5=57.6, U6=89.4.

Отже, таким чином ми встановили ранги всіх вхідних змінних і визначили, що критерій К2 має найбільшу вагу, а критерій К3 - найменшу.

Метод прийняття рішення –метод згортки критеріїв.

Треба визначити найкращий варіант покупки відеогри одного рівня престижності з врахуванням критеріїв: графіка, геймплей, сюжет, звук, атмосфера, оптимізація. Ступень важливості кожного критерію встановлений.

Таблиця 4.3

Відеогра	Геймплей	Графіка	Сюжет	Звук	Оптимізація	Атмосфера
A1	5.5	?	2.5	4	10	7
A2	1	?	4	2.5	1	8.5
A3	2.5	?	1	5.5	4	1
A4	8.5	?	7	8.5	8.5	4
A5	10	?	8.5	1	7	10
A6	7	?	10	10	5.5	5.5
A7	4	?	5.5	7	2.5	2.5
α	0.2	0.2	0.3	0.05	0.1	0.15

Експерти провели порівняльну оцінку графіки відеоігор з метою визначення найкращого результати опитування наведені в таблиці 4.4

Таблиця 4.4

Ранжирування відеоігор по критерію графіки

Експерти	A1	A2	A3	A4	A5	A6	A7
E1	3.5	5	6	1	2	3.5	8
E2	3	5	6	1.5	1.5	4	8
E3	3.5	5	6	1	2	3.5	8
E4	3	5	6	1.5	1.5	4	8

Заданий діапазон значень від $u_{min}(1)$ до $u_{max}(10)$.При чому u_{max} присвоюється найкращому варіанту рішень по кожній властивості (варіанту з мінімальним значенням $X^l j$), u_{min} – найгіршому варіанту рішень по кожній властивості (варіанту з максимальним значенням $X^l j$).Тоді перетворений ранг варіантів може бути визначений наступним чином:

$$Uj^l = U_{\min} + \frac{x_{\max}{}^l - x^l{}_J}{X_{max}{}^l - X_{\min^l}}(U_{\max} - U_{\min}) \qquad (4.2)$$

де $X_{max}{}^l$ та $X_{min}{}^l$ відповідно максимальні та мінімальні сумарні ранги, що отримані при оцінці експертами варіантів по властивості l.

Визначаємо сумарні ранги кожної відеогри:

$X_j1=13$, $X_j2=20$, $X_j3=24$, $X_j4=5$, $X_j5=7$, $X_j6=15$, $X_j7=32$.

За формулою, наданою вище, вираховуємо ранги всіх критеріїв:

$$U1 = 1 + \frac{32-13}{32-5}(10-1) = 7.3$$

U2=5, U3=3.6, U4=1, U5=9.3, U6=6.6, U7=10.

Тепер можемо заповнити нашу таблицю значень всіх критеріїв та їх ваги.

Таблиця 4.5

Таблиця оцінок відеоігор

Відеогра	Геймплей	Графіка	Сюжет	Звук	Оптимізація	Атмосфера
A1	5.5	7.3	2.5	4	10	7
A2	1	5	4	2.5	1	8.5
A3	2.5	3.6	1	5.5	4	1
A4	8.5	1	7	8.5	8.5	4
A5	10	9.3	8.5	1	7	10
A6	7	6.6	10	10	5.5	5.5
A7	4	10	5.5	7	2.5	2.5
α	0.2	0.2	0.3	0.05	0.1	0.15

Проводимо операцію лінійної згортки для прийняття рішення. Спочатку визначаємося з критеріями, діапазоном нормалізованих значень (від 1 до 10), напрямком (максимізація) та вагою критеріїв.

Критерії прийняття рішень

Діапазон нормалізованих значень		1	10	напрям	max
Критерій	Од.вимір.	Напрям	Вага		
Геймплей	Бал	2	0,2		
Графіка	Бал	2	0,2		
Сюжет	Бал	2	0,3		
Звук	Бал	2	0,05		
Оптимізація	Бал	2	0,1		
Атмосфера	Бал	2	0,15		

Рисунок 4.7 Важливість критеріїв

Вводимо значення критеріїв для кожної відеогри (рис 4.8).

Критерійні показники обєктів оцінювання

Обєкти	Значення критеріїв					
	Геймплей	Графіка	Сюжет	Звук	Оптимізація	Атмосфера
Проект 1	5,5	7,3	2,5	4	10	7
Проект 2	1	5	4	2,5	1	8,5
Проект 3	2,5	3,6	1	5,5	4	1
Проект 4	8,5	1	7	8,5	8,5	4
Проект 5	10	9,3	8,5	1	7	10
Проект 6	7	6,6	10	10	5,5	5,5
Проект 7	4	10	5,5	7	2,5	2,5

Рисунок 4.8 Оцінки відеоігор за критеріями

Знаходимо максимальні та мінімальні значення критеріїв, а також визначаємо діапазон значень (рис 4.9).

Максим. Знач. критер	10	10	10	10	10	10
Миним. знач. Критерію	1	1	1	1	1	1
Діпазон значень	9	9	9	9	9	9

Рисунок 4.9 Діапазон значень критеріїв

Проводимо нормалізацію значень критеріїв згідно з напрямом – максимізацією (рис 4.10).

	Проміжні дані для нормалізації значень критеріїв																	
Обєкти	**Відхил.геймплей**			**Відхилення графіки**			**Відхилення сюжету**			**Відхил. Звуку**			**Відхилення оптимызації**			**Відхилення Атмосфері**		
	від максим	від мінім	**прийняте**	від максим	від мінім	**прийняте**	від макси	від мінім	прийняте	від макси	від мінім	**прийняте**	від макси	від мінім	**прийняте**	від макси	від мінім	прийняте
Проект 1	4,5	4,5	**4,5**	2,7	6,3	**2,7**	7,5	1,5	**7,5**	6	3	**3**	0	9	**9**	3	6	**6**
Проект 2	9	0	**9**	5	4	**5**	6	3	**6**	7,5	1,5	**1,5**	9	0	**0**	1,5	7,5	**7,5**
Проект 3	7,5	1,5	**7,5**	6,4	2,6	**6,4**	9	0	**9**	4,5	4,5	**4,5**	6	3	**3**	9	0	**0**
Проект 4	1,5	7,5	**1,5**	9	0	**9**	3	6	**3**	1,5	7,5	**7,5**	1,5	7,5	**7,5**	6	3	**3**
Проект 5	0	9	**0**	0,7	8,3	**0,7**	1,5	7,5	**1,5**	9	0	**0**	3	6	**6**	0	9	**9**
Проект 6	3	6	**3**	3,4	5,6	**3,4**	0	9	**0**	0	9	**9**	4,5	4,5	**4,5**	4,5	4,5	**4,5**
Проект 7	6	3	**6**	0	9	**0**	4,5	4,5	**4,5**	3	6	**6**	7,5	1,5	**1,5**	7,5	1,5	**1,5**

Рисунок 4.10 Нормалізовані значення оцінок

Отже, знаходимо нормалізовані значення критеріїв, а також функцію корисності кожної відеогри (рис 4.11).

	Нормалізовані значення критеріїв						
Обєкти	**Значення критеріїв**						
	Геймплей	Графіка	Сюжет	Звук	Оптимізація	Атмосфера	**Функція корисності**
Проект 1	5,5	3,7	8,5	4	10	7	**6,64**
Проект 2	10	6	7	2,5	1	8,5	**6,8**
Проект 3	8,5	7,4	10	5,5	4	1	**7,005**
Проект 4	2,5	10	4	8,5	8,5	4	**5,575**
Проект 5	1	1,7	2,5	1	7	10	**3,54**
Проект 6	4	4,4	1	10	5,5	5,5	**3,855**
Проект 7	7	1	5,5	7	2,5	2,5	**4,225**

Рисунок 4.11 Комплексна оцінка

На Гістограмах на рис. 4.12 та 4.13 ми можемо побачити, що найкращою відеогрою є Проект 4

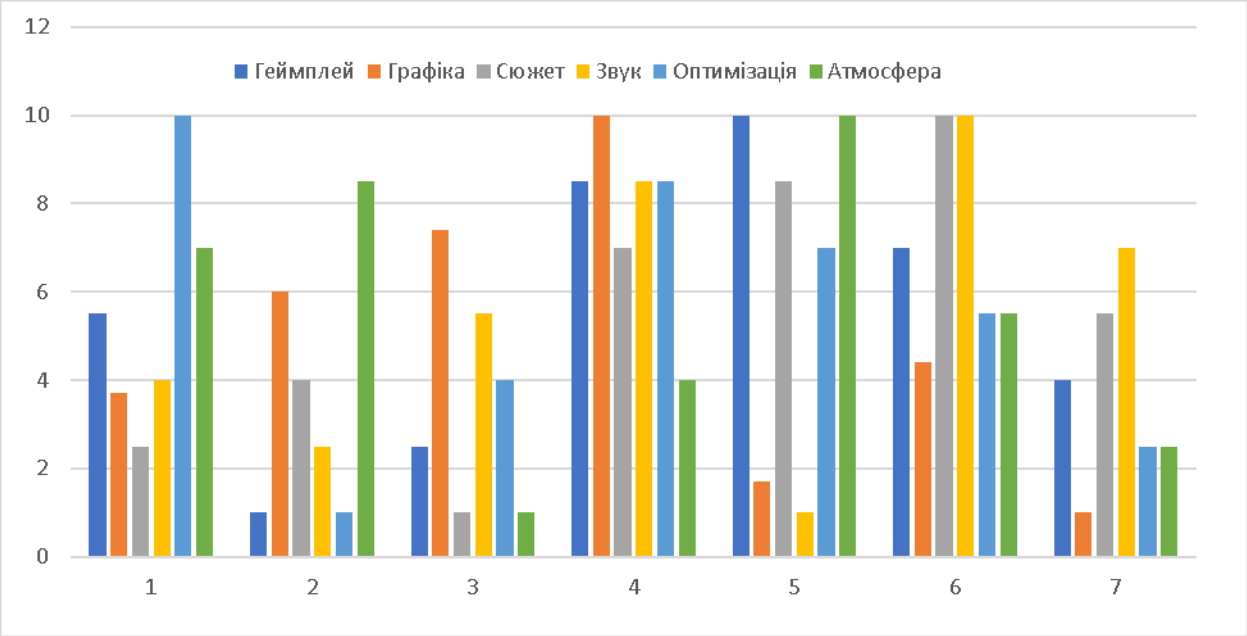

Рисунок 4.12 Гістограма оцінок за критеріями

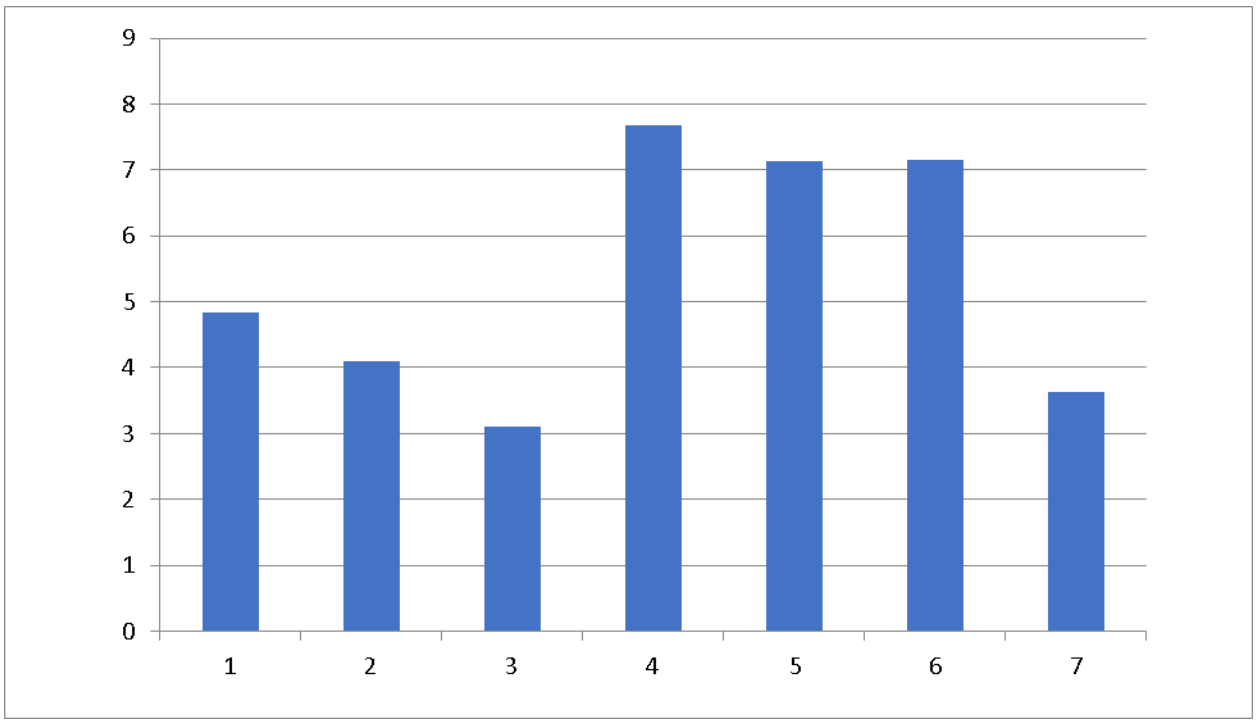

Рисунок 4.13 Гістограма рейтингу відеоігор за якістю

Отже, в результаті оцінки за критеріями створили рейтинг відеоігор та визначили найкращу серед них. Також отримали вагу та міру важливості критеріїв для оцінки відеогри.

ВИСНОВКИ

В результаті атестаційної випускової роботи було виконано:

- Аналіз предметної області. Визначили які спеціальності залученні до розробки відеогри. Розібрали етапи процесу створення відеоігор.
- Створено архітектуру системи розробки відеогри. Були створенні моделі взаємодії її компонентів.
- Застосовано розроблену систему у власному проекті.
- Створено систему оцінки якості відеогри за критеріями. А саме визначили входи та виходи системи.

Дана інформаційна система дозволить уникнути або мінімізувати проблеми під час розробки відеогри. Та підвищити якість продукції завдяки оцінки якості за критеріями.

СПИСОК ВИКОРИСТАНИХ ЛІТЕРАТУРНИХ ДЖЕРЕЛ

1. Video game industry - Statistics & Facts [Електронний ресурс] // Statista. – 2021. – Режим доступу до ресурсу: https://www.statista.com/topics/868/video-games/.
2. A Step By Step Guide on Making a Game Design Document [Електронний ресурс] // TekRevol. – 2020. – Режим доступу до ресурсу: https://www.tekrevol.com/blogs/making-a-game-design-document/.
3. Охріменко В. М. Методичні вказівки до практичних занять з дисципліни „Теорія систем та системний аналіз” / В. М. Охріменко, Т. Б. Воронкова. // ХНАМГ. – 2012. – С. 24.
4. Дерево деталізації функцій [Електронний ресурс] // Stud – Режим доступу до ресурсу: https://stud.com.ua/77210/informatika/derevo_detalizatsiyi_funktsiy.
5. Діаграми UML для моделювання процесів і архітектури проекту [Електронний ресурс] // Evergreen - web розробка і діджиталізація бізнесу за допомогою AI продуктів. – 2021. – Режим доступу до ресурсу: https://evergreens.com.ua/ua/articles/uml-diagrams.html.
6. Згуровський М.З., Панкратова Н.Д. Основи системного аналізу. –К.: Видавнича група BHV, 2007. – 544с.
7. Системний аналіз — Вікіпедія [Електронний ресурс] // Вікіпедія. – 2006. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%BD%D0%B8%D0%B9_%D0%B0%D0%BD%D0%B0%D0%BB%D1%96%D0%B7.
8. Ізмайлова О.В. Методи прийняття багатокритерійних рішень в інформаційних системах: навчальний посібник. – К.:КНУБА, 2002 - 112с.