КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БУДІВНИЦТВА І АРХІТЕКТУРИ
Факультет автоматизації і інформаційних технологій
Кафедра управління проєктами

ПОЯСНЮВАЛЬНА ЗАПИСКА ДО АТЕСТАЦІЙНОЇ РОБОТИ НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ МАГІСТРА

на тему:

Управління проєктом розробки комп’ютерної гри

Степанова Леоніда Павловича
(прізвище, ім’я та по батькові студента повністю)

Київ 2022 р.

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БУДІВНИЦТВА І АРХІТЕКТУРИ

Факультет: Автоматизації і інформаційних технологій
Кафедра: Управління проєктами
Освітній рівень: Магістр за освітньо-професійною програмою
Галузь знань: 12 Інформаційні технології
Спеціальність: 122. Комп'ютерні науки
Спеціалізація: Управління проєктами

ЗАТВЕРДЖУЮ
Завідувач кафедри
Бушуєв С. Д.
«___» _____ 20___ року

З А В Д А Н Н Я

ДО ВИКОНАННЯ АТЕСТАЦІЙНОЇ РОБОТИ НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ МАГІСТРА

Степанова Леоніда Павловича

(прізвище, ім'я та по батькові студента)

1. Тема роботи: Управління проєктом розробки комп'ютерної гри

затверджена наказом ректора КНУБА № 1537/2 від «07» жовтня 2022 року
2. Керівник роботи:

Козир Борис Юрійович, д.т.н., проф.

(прізвище, ім'я та по батькові, науковий ступінь, вчене звання)

3. Строк подання студентом роботи до захисту: 01.12.2022

4. Зміст пояснювальної записки (перелік питань, які слід розробити):
   - теоретичний розділ:
   - дослідження галузі;
   - моніторинг процесів;
   - дослідження з використанням комп'ютерних технологій.

5. Графічний матеріал за розділами: був наданий в роботі відповідно до проведених досліджень та аналізу інформації.

6. Календарний план виконання роботи:

Види робіт та їх зміст	Дата виконання
Збір матеріалів обраного напряму роботи	
Опрацювання та аналіз матеріалів роботи	07.10.22
Вступ	10.10.22 - 20.10.22
Розділ 1.	21.10.22 - 31.10.22
Розділ 2.	01.11.22 - 11.11.22
Розділ 3.	12.11.22 - 22.11.22
Висновки	23.11.22-30.11.22
Остаточне оформлення роботи	30.11.22
Перевірка роботи на плагіат	01.12.22
Попередній захист роботи на кафедрі	06.12.22
Направлення роботи на рецензування	08.12.22

7. Дата видачі завдання _____

Зав. кафедри _____ Бушуєв С. Д.
(підпис) (прізвище та ініціали)

Керівник _____ Козир Б. Ю.
(підпис) (прізвище та ініціали)

Студент _____ Степанов Л. П.
(підпис) (прізвище та ініціали)

РЕЗЮМЕ (summary) *до атестаційної випускної роботи студента:*			**Степанов Леонід Павлович**
ЗВО	Київський національний університет будівництва і архітектури		
Тема	Управління проєктом розробки комп'ютерної гри		
Освітній ступінь	Магістр за освітньо-професійною програмою навчання		
Факультет	Автоматизації і інформаційних технологій		
Кафедра	Управління проєктами		
Спеціальність	122. Комп'ютерні науки		
Спеціалізація	Управління проєктами		
Керівник	Козир Б. Ю., д.т.н., проф.		
Обсяг роботи:	*пояснювальна записка, сторінок*	*розділів*	*слайдів презентації*
	94	*3*	*15*
Розділ 1. Розважальне програмне забезпечення. Основні положення. Аналіз ігрової індустрії	У першому розділі був розглянутий продукт комп'ютерних відоігор, стан сучасної ігрової індустрії та дана невелика історична довідка для розуміння контексту галузі. Для розуміння середовища для виконання цього проєкту був проведений огляд вітчизняної сцени та проведені порівняння стану зростання цінності провідних відеоігрових студій відносно великих компаній більш традиційних галузей. Аналіз ситуації показав, що в Україні не присутня велика кількість студій, які розробляють продутки такого типу. Беручи до уваги факт того, що існують місцеві офіси зарубіжних компаній як Ubisoft чи Plarium показує, зрозуміло, що Україна має достатній вітчизняний ресурс спеціалістів для розробки розважального програмного забезпечення.		
Розділ 2. Дослідження методів розробки	Другий розділ присвячений вивченню інструментів для реалізації процесу розробку програмного продукту за		

ігрових продуктів. Розповсюджені фреймворки та практики у сфері інформаційних технологій	допомогою управління проєктами. У ньому були розглянуті сучасні методології управління проєктами в галузі інформаційних технологій. Було проведене дослідження опцій для розробки програмного забезпечення. У результаті огляду інструментів, доступних для галузі та аналізу успішних проєктів, схожих до розроблюваного, були визначені фактори, за допомогою яких буде легше досягти цільової аудиторії і розробити успішний продукт.
Розділ 3. Реалізація управлінням процесу розробки "Runic Forest". Оформлення проєктної документаці	У третьому розділі був детально розглянутий процес розробки програмного забезпечення "Runic Forest". У результаті розробки статуту були визначені унікальні сторони проєкту. Були встановлені вимоги, розроблені завдання та список їх виконавців, прораховані ризики, витрати та створена система гнучного менеджменту ресурсів.
Висновки по роботі:	В атестаційній роботі на здобуття освітнього ступеня магістра був проведений глибокий аналіз інструментів для управління проєктами та був застосований в сфері планування процесу розробки програмного забезпечення. У результаті проведення дослідження вітчизняної галузі і особливостей проєкту, була обґрунтована необхідність розробки програмного забезпечення "Runic Forest". Був створений план управління проєкту, у якому були розглянуті принципи взаємодії різних ресурсів проєкту та був проведений аналіз заходів для забезпечення успішності виконання завдання по розробці програмного продукту. Також були створенні схеми управління ресурсами, ризиками, вартістю, термінами та

	комунікаціями в рамках проєкту.
Ключові слова: розробка відоігор, проєкт, програмний продукт, управління проєктами, застосування інформаційних технологій. **Keywords:** development of videogames, project, software product, project management, application of information technologies.	

Укладач: Степанов Л. П.

Керівник: Козир Б. Ю.

«___» _____ 2022 р.

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БУДІВНИЦТВА І АРХІТЕКТУРИ

Факультет автоматизації і інформаційних технологій

Кафедра управління проєктами

ЗАТВЕРДЖУЮ
Завідувач кафедри
Бушуєв С. Д.
«___» _____ 20___ року

ПОЯСНЮВАЛЬНА ЗАПИСКА
ДО АТЕСТАЦІЙНОЇ РОБОТИ НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ МАГІСТРА

Управління проєктом розробки та комп'ютерної гри

(назва)

Виконав студент групи: УП-61
Степанов Леонід Павлович
(прізвище, ім'я та по батькові повністю)

Спеціальність: 122. Комп'ютерні науки
Спеціалізація: Управління проєктами

Керівник: Козир Б. Ю.
(прізвище, ініціали,)
д.т.н., проф.
науковий ступінь, вчене звання
Рецензент: _____
(прізвище, ініціали,)
_____
посада

Київ 2022 р.

ЗМІСТ

ВСТУП

Управління проєктами виникло в час, коли людям вперше знадобилися методи керування командами для досягнення результатів, на які не була би спроможна одна людина. У нашу еру людського капіталу і гігантських корпорацій, які можуть об’єднувати людей по всьому світу, як користувачів, так і тих, хто працює над їх продуктами, в велику мережу комплексної взаємодії. Для того, щоб все було під контролем, людство відпрацювало на своїх помилках та перемогах досить великий базис для дослідження процесів створення нового та його підтримкою. Серед замовників, інвесторів, консультантів і технічних робітників існує робота проєктного менеджера. Вона покликана максимально спростити процеси організації роботи та комунікації між відділами. Тобто, можна сказати, що проєктний менеджер є серцем будь-якого процесу розробки та підтримки продукту, від зачину проєкту і аж до кінця циклу розробки, після якого йде новий предмет дослідження.

На сьогоднішній день більшість проєктних менеджерів працюють в сфері інформаційних технологій. Згідно з даними щорічного звіту за 2020 рік профільної консалтингової компанії, яка базується в Великобританії, Wellingtone, галузь IT займає перше місце за кількістю продуктів, над якими працюють проєктні менеджери, обганяючи індустрію освіти, консультаційних послуг та виробництва – важкої і легкої галузі [1].

Одним з найбільш уживаніших програмних продуктів в сучасному світі є комп’ютерні ігри — тобто такі, які забезпечується програмно керованим електронним пристроєм — комп'ютером [2]. На сьогоднішній день можна з впевненістю сказати, що ігрова індустрія сформувалася як досить самостійна та прибуткова галузь. Для дизайну розважального програмного забезпечення використовують стандартні шаблони для створення будь-яких інших програмних продуктів. Результатом розробки є продукт, який задовольняє певну потребу. Проте існує деяка кількість девіацій в цьому процесі, які виділяють саме

проєктування саме відеоігор на фоні іншого програмного забезпечення. Для розробки розважального програмного забезпечення в кожній студії існує свій власний підхід, який був встановлений в процесі формування політики та культури середовища кожного з них.

Для створення продукції будь-якому підприємству необхідні фінансові ресурси, які б забезпечували його функціонування. Ця аксіома стосується і ігрових студій. З пошуку таких джерел починається розробка нового проєкту як для великих, так і для маленьких студій. Індустрія відеоігор характеризується як класичними джерелами фінансування, які використовуються підприємствами всіх сфер, так і особливими (унікальними) джерелами, які не можуть бути застосовані для підприємств інших галузей. Компанії, що займаються відеоіграми, працюють на комерційній основі, тому прагнуть максимізувати прибуток. Для цього після закінчення розробки кожна компанія-розробник обирає найбільш ефективну модель монетизації свого продукту, її вибір залежить від специфіки розробленої гри [3].

Станом на 2021 рік в Україні налічувалося 80 ігрових студій. Серед них велика кількість малих команд, які працюють над іграми на мобільному ринку та аутсорсі (віддаленій роботі на замовлення) для іноземних студій. В країні існує великий потенціал для того, щоб розвивати різні галузі IT, включаючи сферу розважального програмного забезпечення.

Для того, щоб задовольнити інформаційні потреби буде проведено планування та дослідження процесу створення відеогри, на прикладі “Runic Forest”. Для цього були заплановані та поставлені подальші задачі:

1. Провести аналіз принципів та сталих в процесі розробки розважального програмного забезпечення;
2. Провести дослідження тенденцій внутрішнього ігрового ринку України та дослідити успішні світові практики;
3. Визначити специфічні ознаки роботи в галузі та імплементувати;
4. На основі отриманих даних визначити цінності та методологію для реалізації проєкту;

5. Створити план та проєктну документацію, за якою буде проведена робота над проєктом.

Предмет дослідження: ігрова індустрія України та методики розробки комп'ютерних ігор

Об'єкт дослідження: процес розробки комп'ютерної відеогри "Runic Forest"

Теоретична та методологічна основа дослідження: для роботи над предметом дослідження були використані вітчизняні та іноземні статті, статистичні звіти галузевих компаній. Також використані витримки з різних професійних джерел, таких як PMBOK та Agile Alliance, та доповіді працівників лідерів галузі на тематичних професійних конференціях.

Практичне значення: основною метою проведеної роботи повинні стати загальні вказівки та практична база створення розважального програмного забезпечення. Також, робота покликана вивчити сучасні методи їх розробки, створення оформлення та просування, що допоможе галузі набрати обертів на вітчизняному ринку та вивести країну на більш конкурентний рівень на світовому ринку.

Склад роботи: робота складається зі вступу, основної частини в трьох розділах, висновків, списку джерел. Робота включає 7 рисунків і 31 літерітурне джерело.

РОЗДІЛ 1. РОЗВАЖАЛЬНЕ ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ. ОСНОВНІ ПОЛОЖЕННЯ. АНАЛІЗ ІГРОВОЇ ІНДУСТРІЇ

1.1. Історія ігрової індустрії

1.1.1. Витоки історії ігор

Історія ігор починається з давніх часів. Найстарішою археологічною знахідкою, яку було ідентифіковано як гру, вважають древню настільну гру “сенет”, в яку грали люди зі свити та сімей династій фараонів більш ніж 5500 років тому у Древньому Єгипті. Вона складається з невеликої коробки з ігровим полем згори з різноманітними фігурками, які в ній і зберігаються. Точні правила гри невідомі, проте це показує, що ігри, як спосіб рекреації, з’явилися задовго до історії відомих нам на даний час. Вчені припускають, що деякі, навіть більш давні знахідки, можуть бути ігровими костями, що датуватиме перших представників цього виду розваг ще давніше.

Свої ігри існували фактично в усіх розвинених древніх цивілізаціях, як Єгипет, Індія, Китай, цивілізації корінних жителів Америки, Міжріччя. Майже усі вони складалися з певного поля та рухомих фігур, які повинні рухатися через поле. Перші ігри були майже виключно розраховані більше, ніж на одного гравця, тому що часто за допомогою них проводили азартні змагання. Один із гравців перемагає, коли досягає певної точки, до якої йдуть усі учасники за допомогою дій, прописаних в правилах. Це надавало їм величезної варіації. Чим складніше було поле гри і чим комплекснішими були її правила, тим більш варіативним був процес. Це один із факторів, який треба брати до уваги, створюючи будь-яку ігрову систему. Незважаючи на те, як багато часу пройшло з винаходу перших, сучасні представники, настільні чи комп’ютерні, керуються логікою варіативності.

Сучасна історія ігор бере початок з перших професійних турнірів, тобто з початком встановлення універсальних правил, першим представником яких стали

шахи. Сучасні правила шахів почали формуватися в Іспанії та Італії 15 століття з прийняттям стандартних паттернів ходів тури та королеви, які спочатку називалися «Шахами божевільної королеви». Твори з теорії шахів також почали з'являтися в 15 столітті. Першим текстом був «Repetición de Amores y Arte de Ajedrez» іспанського церковника Луїса Раміреса де Лусено. Розвиток літератури сприяв розповсюдженню практики універсальних правил. Книги про шахи таких авторів, як Руй Лопес де Сегура та Джоакіно Греко, стали широко вивчатися.

З цим почала розвиватися комерційна частина індустрії. Починаючи з часу індустріальної революції та автоматизації виробництва, вона стала складовою життя більшості людей робочого класу більш-менш розвинених країн. Поля та аксесуари для ігор почали приймати універсальні форми, з якими ми знайомі сьогодні.

1.1.2. Сучасна історія ігрових продуктів. Електронна трансформація.

З розвитком обчислювальних машин, вони ставали все дедалі більш багатофункціональними. З простих логічних операцій вибудовувалися системи складних інтеракцій між фізичними схемами взаємодії контактів.

Перші представники цього класу програм з'явилися це на етапі початку розвитку електронних обчислювальних машин, проте їм передували суто машинні рішення, які працювали за алгоритмами, записаними на рівні власне мікросхем. Найстарішим продуктом, який прийнято називати комп'ютерною відеогрою, вважають "Bertie the Brain"[4]. Це була машина, яка була побудована для проведення сесій в tic-tac-toe. Це досить проста розвага, та усі ходи можна прорахувати заздалегідь. Сам продукт для неї складався з простого монітору та досить великої електронно-обчислювальної машини, яка прораховувала кроки гравця, тобто людини, від якої потребується ввід даних для того, щоб гра працювала. Незважаючи на те, що головною метою винаходу була розважальна функція, проте справжньою причиною для показу цього приладу була спроба

прорекламувати практичним та зрозумілим шляхом новий логічний елемент додавання. Тож, можна зазначити, що історія комп'ютерних ігор ще з самого початку йде зі спроб введення інновацій. Проте публіка швидко забула про цей винахід, коли ефект культурного шоку скінчився.

Найпростіші ходи, які можна передбачити через їх невелику варіативність та простоту гри, були під силу мікросхемі з ламп.

Відеогра PONG, яка не була першою, вважається першим представником того, що мають на увазі, коли вживають слово "відеогра". 29 листопада 1972 року в таверні Енді Кеппа в Саннівейлі, Каліфорнія, була машина, сконструйована Елом Алкорном, інженером, який працював на ігрових підприємців Нолана Бушнелла та Теда Дабні, які нещодавно об'єдналися під назвою «Atari»[5]. Прилад мав чорно-білий екран, на якому було дві платформи, кожну з яких контролював один з гравців. Останнім елементом ігрового поля був "м'яч", який відбивався від цих платформ. Процес повинен симулювати гру в настільний теніс, від якого гра і бере свою назву. М'яч безперервно рухався, і якщо він пролітав повз платформу гравця, його опонент отримував бали. Перемагав гравець, який пропустив найменшу кількість голів. Це був перший комп'ютер з класу, котрий буде прийнято називати "аркадними автоматами".

Тобто комп'ютерна система надала можливості, які неможливо було реалізувати без нього. Сам концепт був надто абстрактним для реалізації у будь-якому іншому вигляді. Це другий фактор, який робить відеоігри продуктом, на який є зиск – ексклюзивність ігрового процесу.

З розвитком аркадних машин почали з'являтися зменшені варіанти приладів, які дозволяли людям грати в ті самі ігри з комфорту їх власної оселі. Приладом виводу візуальної інформації були телевізори, до яких вони підключалися. Зрештою були розроблені системи, які могли запускати ігри з зовнішніх носіїв, таких як картриджі. Саме тут і почалася історія комерціалізації саме відеоігор, як продукції. Для розробки розважального контенту не треба було з нуля будувати новий прилад, а можна було розробити картридж, який все існуюча машина інтерпритує. Все починалося з того, що виробник машин

виробляв й картриджі, що давало їм повну монополію на систему. Проте з часом з'являлися підрядники, які продавали свої ігри, віддаючи частину прибутку виробнику консолі.

Наступною віхою розвитку стали повноцінні домашні комп'ютери, які дозволили принести складні мови програмування до мас. З часу перших домашніх машин, була розроблена велика кількість ігор – написаних на комп'ютерах, для комп'ютерів.

1.1.3. Психологія інтерактивного розважального контенту

Для розуміння цінностей, які будуть важливі в процесі виконання проєкту, звернемося до психологічної складової ігор як явища. Дослідивши їх, можна буде розставити пріоритети під час процесу розробки продукту. Серед причин, чому люди грають у відеоігри компанія, яка досліджує мотивацію користувачів, Immersyve, наводить три основні:

- **Компетентність** – це перша причина. Нам подобається відчувати, що ми в чомусь хороші, і нам подобається, коли нас за це визнають. Ми хочемо знати, що ми впоралися з ситуацією, і нам подобається відчуття прогресу та досягнення цілей. Це вірно в житті і проявляється як наше бажання зробити кар'єру, отримати підвищення або змінити роботу, або зайнятися новим хобі чи навчитися чогось нового. Це вбудовано в ігри. Вони пропонують виклики різного ступеня складності з чіткими лініями прогресування. Вони також надають нам вбудовані системи винагород.
- Друга психологічна потреба, до якої звертаються ігри, - це **автономія** або наше прагнення до незалежності. Ми хочемо відчувати, що контролюємо свої дії та ситуації. Однак у реальному житті це не завжди легко. Багато чого може статися поза нашим контролем, і це може засмучувати; прикладом цього може бути поглиблений досвід гри віртуальної втечі. Ігри з вільним переміщенням, такі як GTA, особливо добре пропонують автономність, оскільки гравець може майже самостійно прокладати собі

шлях у грі. І хороша новина щодо ігор полягає в тому, що невдача не коштує нам нічого. Гравці можуть зазнати невдачі та спробувати знову, і все це без зайвого ризику.

- **Відносини** – це третя психологічна потреба, до якої звертаються ігри. Нам подобається відчувати, що ми маємо значення для інших людей і що ми змінюємо нашу групу чи суспільство. Багатокористувацька гра, а особливо масові онлайн-ігри, забезпечують цей зв'язок у дуже прямий спосіб, але дослідження Immersyve показали, що ми навіть ставимося до вигаданих персонажів у грі та відчуваємо спорідненість через діалоги та квести, щоб допомогти іншим.

Люди купують ігри для задоволення саме цих потреб. Різні жанри ігор задовольняють одну або декілька з них. Станом на наш час існує велика кількість різноманітних типів ігор, які відрізняються тим, як саме гравець досягає точки перемоги та що він робить, щоб це зробити. Серед основних та розповсюджених ігрових жанрів існують такі:

- **Екшн-ігри** є одним із найпопулярніших жанрів відеоігор у світі, і зазвичай містять фізичні випробування, які долаються стрибками, бійками та стрільбою. Відмінна координація рук і очей і рефлекси посмикувань зазвичай потрібні в екшн-іграх, а дія зазвичай обертається навколо персонажа користувача, який повинен пройти серію випробувань, щоб перейти до наступного рівня.
- **Екшн-пригодницькі ігри містять** елементи як екшену, так і пригодницької гри. Багато аспектів пригодницьких ігор подібні до ігор бойовиків, але вони також пропонують сюжетну лінію, персонажів, систему інвентарю, діалоги та інші особливості пригодницьких ігор. Пригодницькі ігри зазвичай включають комбінацію складних сюжетних елементів, які відображаються гравцям за допомогою аудіо та відео. Історія значною мірою залежить від дій персонажа гравця, які запускають події історії та таким чином впливають на хід гри.

- **Пригодницькі ігри** – це ігри для одного гравця з захоплюючим сюжетом. Більш ніж будь-який інший жанр відеоігор, ці ігри значною мірою залежать від свого сюжету та сценарію, щоб забезпечити захопливий досвід для одного гравця. Головоломки, розшифровка повідомлень, пошук і використання інструментів, відкриття замкнених дверей, а також відкриття та дослідження нових місць — усе це частина пригодницьких ігор. Розгадування головоломки дозволяє досліджувати нові регіони ігрового середовища та більше дізнаватися про сюжет. Інтерфейс керування запасами — це особливий стиль гри в багатьох пригодницьких іграх. Оскільки гравці можуть підбирати лише певні об’єкти в грі, вони зазвичай розуміють, що лише ті об’єкти, які можна підібрати, є важливими.
- **У рольових іграх** гравець бере на себе роль персонажа, у якого зростає сила та досвід протягом гри. Від гравця вимагається подолати складні випробування та/або перемогти монстрів, отримати очки досвіду, які відображають прогрес персонажа у вибраній професії чи класі та дозволяють гравцеві отримати нові здібності після отримання встановленої кількості.
- **Стратегічні ігри** – це популярні жанри відеоігор, у яких наголошується на мисленні та плануванні, а не на прямих миттєвих діях для перемоги. Гравець повинен виконати серію дій проти одного або кількох супротивників, щоб зменшити кількість військ супротивника. Перемога досягається шляхом більшого планування, а удача відіграє другорядну роль. У більшості стратегічних відеоігор користувачеві надається божественний погляд на ігровий всесвіт і опосередковано керує ігровими одиницями. Як наслідок, більшість стратегічних ігор різною мірою включають тактичні та стратегічні фактори, а також елементи війни. Ці ігри часто перевіряють здатність гравця досліджувати економіку та керувати нею, окрім боротьби.
- **Ігри-головоломки** перевіряють навички гравця розв’язувати проблеми, вимагаючи від нього розпізнавати шаблони, розгадувати послідовності та завершувати слова. Кросворди, головоломки для пошуку слів, числові

головоломки, реляційні головоломки та логічні головоломки є прикладами різних жанрів головоломок. Головоломки часто розробляються для розваг, але вони також можуть бути результатом значних математичних або логічних проблем. Логічні та концептуальні задачі є головною темою головоломок. У той час як багато бойовиків і пригодницьких ігор включають аспекти головоломки в дизайн рівнів, справжня гра-головоломка надає першочергове значення розв'язанню проблем як основному ігровому процесу.

За допомогою жанрів, користувач може підібрати те, що йому подобається за списком ярликів та стереотипів в ігровому процесі. Тому треба орієнтувати роботу на тому, на чому зможуть зосередитися члени команди та плідно працювати. Так як у наш час легко знаходити ігри, які користувач знайде цікавими, розробникам залишається зробити правильні рішення, які зроблять так, що аудиторія для розроблюваного продукту зможе побачити гру, яка задовольнить їх рекреаційні потреби.

1.1.4. Монетизація. Проблеми і способи їх вирішення

Створення відеоігри, як і створення фільму, може бути надзвичайно дорогим. Крім того, гравці очікують, що сьогодні більшість ігор будуть безкоштовними, а це означає, що студії повинні бути креативними, щоб отримувати дохід. Будь-яка система монетизації створюватиме тертя для користувача. Зрештою, витрачання грошей зазвичай не сприймається як хороший досвід. Ігри, які мають дивовижну рекламну кампанію, яка спонукає захоплених геймерів платити за можливість отримати доступ до гри, але потім розчаровуються, оскільки гра не відповідає обіцяним неймовірним враженням, обманюють. Перевага безкоштовних ігор полягає в тому, що гравці можуть спробувати гру безкоштовно. Їм не потрібно вірити студії на слово і купувати гру, вірячи, що рекламований досвід саме той, який вони отримають. Але це створює нові типи точок тертя для гравців і новий потенціал для «темних шаблонів».

Темний шаблон — це дизайн, який навмисно вводить в оману, з кінцевою метою принести користь компанії за рахунок користувачів. Але з безкоштовними іграми (та додатками) з’явилися нові темні шаблони. Наприклад, вони можуть приймати форму певного тиску на гравців, щоб вони брали участь у грі певний час або в певний день, інакше вони пропустять або щось утратять. Це називається страх втрати, і його можна використовувати як темний шаблон, щоб вплинути на те, щоб гравці грали щодня, інакше вони можуть втратити нагороду, яка їм важлива (цю техніку, разом з іншими, часто вважають частиною «економіки уваги»). Ця механіка не була винайдена індустрією відеоігор і, звичайно, використовується не лише цією індустрією. Роздрібна торгівля використовує страх втрати протягом десятиліть, наприклад, коли повідомляє клієнтам, що вони можуть втратити чудову економію, якщо не купуватимуть товар у певний день за допомогою сезонних чи обмежених знижок та спеціальних пропозицій.

Як і з іншими методами отримання прибутку, необхідно дати споживачу чітке розуміння, за що саме він віддає гроші та що він отримує роблячи це. Проте, незважаючи на те, що ми живемо в світі, орієнтованому здебільшого на споживачів, потрібно, щоб в результаті команда отримувала прибутки за свою роботу. Часто ці два фактори балансують один між одним, залишаючи щасливих користувачів, задоволених інвесторів і ситих робітників. Це базовий принцип “доброї волі”, або його ще прийнято називати “goodwill”.

Вище згадані темні шаблони мають змогу принести велику кількість прибутку за невеликий проміжок часу через психологічні маніпуляції. Проте, якщо гравець не буде співставляти для себе цінність, яку він отримує, з використаними грошами, використовуючи азарт користувача, якщо гра має елементи таких ігор, або маніпулючи його бажаннями, згаданими вище: забирати повністю або частково компетентність, автономію, чи соціальний аспект. Такі практики є неетичними та створюють поганий репутаційний фон для студії розробника.

Тож найкращий варіант для підтримки здорового ринку та довіри користувача до постачальника залишається прозора система, де він чітко розуміє,

скільки саме грошей йому треба вкласти в гру, щоб отримати повний досвід, яким його задумували дизайнери гри.

Існує велика кількість систем монетизації, на які спираються компанії для отримання прибутку [6]. Серед них виділяють основні види, наведені в таблиці 1.1.

Таблиця 1.1

Джерела фінансування ігор

Вид джерела фінансування	Короткий опис принципу	Сфера використання
За рахунок власних коштів	Проєкт фінансується напряму з коштів студійного бюджету	Як великі, так і починаючі студії
За рахунок кредитних коштів	Проєкт фінансується з кредитних коштів	Не є поширеним так як часто не перекриває всі витрати на проєкт
За рахунок інвестицій	Проєкт фінансується за рахунок зовнішніх фінансових надходжень з боку інвесторів	Великі компанії з великим досвідом
За рахунок видавця	Проєкт фінансується видавцем, який задає курс та бачення гри, тобто по суті є свого роду просунутим інвестором	Як великі, так і малі студії
За рахунок краудфандингових коштів	Проєкт фінансується зі сторони зацікавлених користувачів через збір коштів на ранньому етапі розробки	Невеликі студії, часто з вузьким профілем, розрахований на ще не зайнятий ринок

Закінчення табл. 1.1

За рахунок інвестицій платформи	Проєкт фінансується або компаніями платформ для ігор, або ігрових магазинів, часто з ексклюзивністю для цієї платформи, як основною вимогою	Як великі, так і малі студії. Частіше всього великі незалежні студії отримують гранти за ексклюзивність
За рахунок передпродажів	Проєкт фінансується за рахунок купівлі користувачем не до кінця завершеної гри, яка має реалізовану основну ідею фінального продукту, геймплейної складової	Невеликі студії. Часто за рахунок передпродажів намагаються визначити, чи зацікавить гра цільову аудиторію

Серед цих методів існують як і традиційні (кредитні кошти та інвестиційні надходження), так і специфічні для галузі (через фінансування платформами, видавцями, чи навіть самою аудиторією). Ці методи мають свої сильні та слабкі сторони, які представлені в таблиці 1.2.

Таблиця 1.2

Переваги та недоліки методів фінансування

Вид джерела фінансування	Переваги методу	Недоліки методу
За рахунок власних коштів	Творча свобода	Обмеженість коштів, потреба в початковому фінансуванні
За рахунок кредитних коштів	Творча свобода і фінансовий достаток	Кредитні зобов’язання

Закінчення табл. 1.2

За рахунок інвестицій	Творча свобода і фінансовий достаток	Можливість втрати інвестицій
За рахунок видавця	Фінансовий достаток	Зменшення прибутку, творча обмеженість
За рахунок краудфандингових коштів	Творча свобода, повний прибуток	Відсутність гарантій фінансування
За рахунок інвестицій платформи	Фінансовий достаток, забезпечення технологіями платформи	Зменшення прибутку, творча обмеженість
За рахунок передпродажів	Творча свобода	Потреба в хоча б частково готовому продуті

1.2. Комп'ютерні ігри як продукт. Сильні і слабкі сторони

Розробка розважальних програм має як і переваги над виробництвом інших видів медіа, так і недоліки, або слабкі місця. Відеоігри є інтерактивним досвідом, що сильно виділяє його на фоні інших способів розповіді певної історії або донесення певних ідей та емоцій до споживачів.

Серед переваг відеоігор над іншими медіа-продуктами:

1. **Інтерактивність**. Основною причиною для реалізації певної ідеї у вигляді інтерактивного медіа є досвід, який буде персоналізованим кожним користувачем для нього самого. Також це має ефект причетності – часто гравці мають можливість використати особливий підхід до проходження продукту, що підвищує рівень іммерсії та покращує наративні елементи, які присутні в грі.

2. **Реграбельність** – це можливість пройти гру іншим чином. Деякі з них дають більше вибору, інші ж – менше, проте це є однією з причин вибрати саме цей медіум для розповіді певної історії, особливо нелінійної, або такої, яку робить сам гравець.
3. **Масовий мультиплеєр**. Існує велика кількість відеоігор, який заснований на принципі урахування великої кількості вводів даних великою кількістю гравців. Це дає змогу участі в одному процесі надзвичайно великої кількості учасників, враховуючи те, що вони б не змогли зібратися у фізичному просторі за такої кількості.
4. **Легкий доступ користувача та скалювання**. На сьогоднішній день легко знайти місце для того, щоб викласти гру, де її побачать. Такі сервіси, як Steam, Epic Games Store та GOG, дозволяють викласти продукт за певну плату на їх сервісах, та підтримувати ці сторінки за рахунок комісій з продажів. У той час користувачам легко використовувати сучасні пошукові по тегам та жанрам, що допоможе розробникам знайти свою аудиторію.

Серед недоліків відеоігор відносно інших медіа-продуктів:

1. **Потреба в ігровому пристрої.** У більшості людей сьогодення є доступ до телефонів, які у тисячі разів більш продвинуті, ніж багатокоміркові комп’ютери минулого, які займали цілі кампуси і коштували сотні тисяч, а сам комп’ютер чи приставка стали звичними приладами для дому. Проте факт залишається фактом. Відеогра – не самостійний продукт, і обмеження по потужності приладу, його архітектури та операційної системи роблять ці продукти куди менш універсальними для споживання, ніж музика чи фільми.
2. **Час розробки інновацій.** З більш комплексними методами розробки та технологіями стає все складніше вигадувати нові механіки, незважаючи на те, що самих інструментів для розробки стає більше. Розробка гри – це в першу чергу креативний процес, і технічні деталі можуть затримати процес досить надовго.

3. **Залежність від технологій.** Переходи на нові види операційних систем, серед програмування та ігрових рушіїв, які постійно оновлюються, стають великою перепоною для планування процесів розробки.

1.3. Стисла характеристика сучасної ігрової індустрії

Індустрія відеоігор охоплює розробку, маркетинг і монетизацію відеоігор. Виходячи з вище наведеної історичної довідки можна побачити, що ігри існували ще з часів початку людської цивілізації, а електронні ігри існували з часу розквіту інформаційних технологій. Перші представники других брали своїм фактором новизни, проте. Коли перші підприємці почали виробляти ігрові продукти, вже через декілька років нішовий бізнес вибухнув в справжню індустрію.

Наразі все більше інвесторів почуваються комфортно, коли вони вкладають свої кошти в віделігрові бізнеси. Так як це суто продуктова газуль, багато факторів успішності інвестицій залежить від того, як себе покаже продукт, тобто це стосується більше великих гравців на ринку, проте це стає трендом.

Сучасна ігрова індустрія характеризується такими ознаками:

- Ігрова індустрія має дві сцени: видавницьку і незалежну. Видавці є великими гравцями в сфері
- Як і будь-яка інша сфера IT, ігрова індустрія швидко розвивається і швидко змінює свої тренди
- Не існує чітких алгоритмів роботи над продуктами, кожна студія має свої процеси розробки.

Для того, щоб дати відповідь на питання доцільності інвестицій в дану галузь, були взяті графіки швидкості приросту акцій для порівняння того, як швидко розвиваються ігрові студії відносно представників інших, більш традиційних галузей. Для прикладу були взяті результати порівняння найбільших компаній США (S&P500, Activision Blizzard) (рис. 1.1.), як представника

передової економіки та сусідньої Польші (WIG20, CDPR), з економікою, яка розвивається (рис. 1.2.).

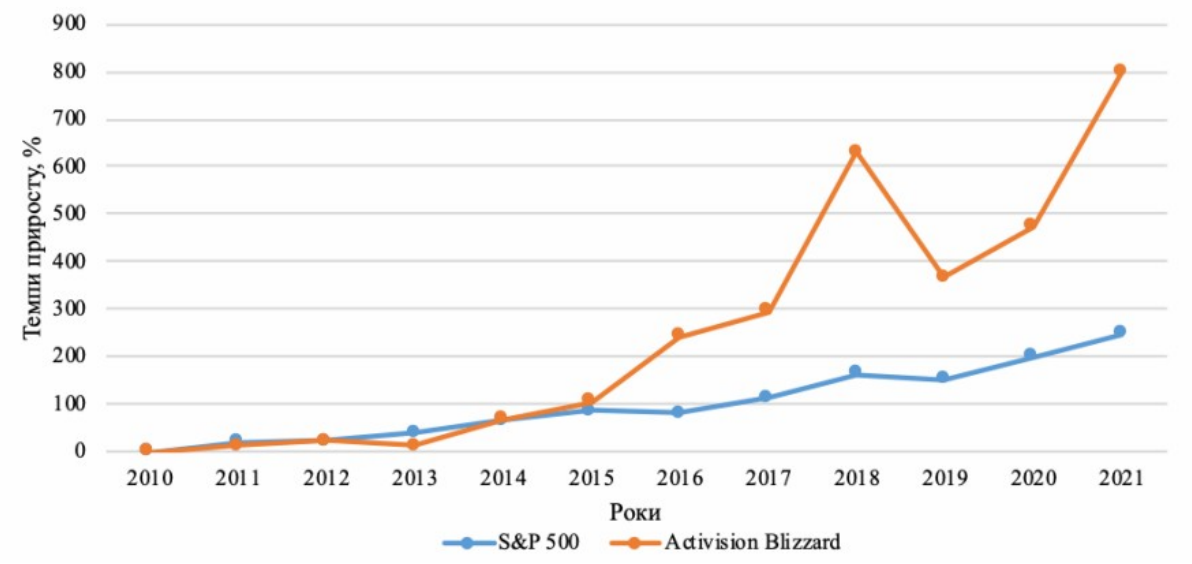


Рис. 1.1. Темп приросту акцій S&P500 і Activision Blizzard

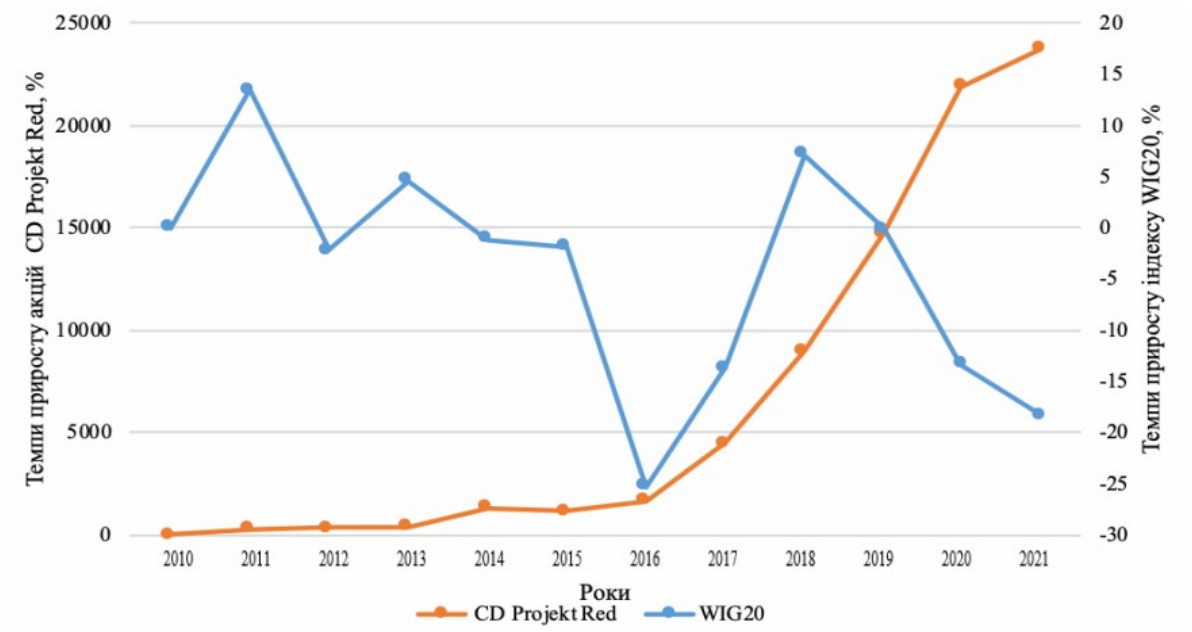


Рис. 1.2. Темп приросту акцій WIG20 і CDPR

Виходячи з графіків, існують бурхливі тенденції розвитку галузі. Передові представники галузі швидко набирають обертів в цінності їх акцій. Компанії-виробники задовольняють потреби ринку, який розширюється високими темпами.

Як результат розвитку галузі, в ній спостерігаються такі тенденції:

- Стрімке зростання прибутковості галузі

- Розширення сфер використання геймифікації та ігрових методів для галузей освіти, медицини, соціальних додатках
- Збільшення кількості людей в командах проєкту, створення розгалуджених систем менеджменту
- З розвитком апаратного забезпечення збільшується і комплексність ігрових продуктів
- З кожним днем стає все більше людей, які грають в хоча б один вид відеоігор, тобто швидко збільшується і користувацька база

Після процесу розробки ігри, як і будь-який інший медійний продукт, треба видавати. Ігрова індустрія характеризується великою кількістю компаній, які цим займаються. Більшість з них починали свій шлях зі стану студій розробки, проте, створивши достатньо продуктів та після отримання достатнього фінансового плацдарму, вони переходили до виробництва. Через те, що великі команди потребують більше персоналу з “м’якими навичками”, які працюють для того, щоб забезпечувати вони мають більше можливостей для розміщення, просування та менеджменту продукту. Також, компанії, які довго протрималися на ринку, часто мають деяку базу лояльності, яка допоможе просувати продукти, створені менш відомими командами.

В індустрії розповсюджені такі практики взаємодії видавця і студії:

- Менторство
- Видавництво
- Поглинання

В першому випадку менторства, видавець надає працівників або деякий інший ресурс для допомоги команді. У світі ігрової індустрії досить поширена практика обміну досвідом серед експертів галузі. Так як ігри є продуктом досить артистичним, нерідко хороша концепція, створена невеликою студією, може зацікавити гігантів жанру. У кінцевому випадку найголовніше, чого потребує як виробник, так і споживач – це якісний кінцевий продукт, який стає більш можливим з допомогою більш досвідчених експертів.

В випадку видавництва, студія звертається до видавництва за допомогою в просуванні та розміщення продукту на одній з платформ, керування її рекламною компанією за частку прибутку від продукту. Часто в такому випадку гра вже існує в деякому вигляді, або вже є завершеною, а видавець слугує гарантом для того, шоб продукт побачив свого потенційного споживача.

Студії, які вже зарекомендували себе як успішні часто викупаються і переходять під крило видавництва для посилення їх бренду. У такому випадку команда отримує стабільне фінансування, певний платіж в бюджет, проте втрачає частину або повну свободу у процесі визначення ігор для розробки, їх ідей та використання технологій. Деякі видавці, вихідці з таких студій, створюють свої інкубатори та системи менеджменту, щоб протидіяти такому підходу та профілювати нові та експериментальні проєкти, як наприклад Devolver Digital і Chucklefish.

1.4. Стан ігрової індустрії в Україні. Тенденції та перспективи розвитку

У вітчизняній сфері ігрової індустрії ситуація досить складна. Згідно з докладом опитування української компанії-технологічного парку UNIT, в Україні діє велика кількість офісів іноземних ігрових гігантів, як Ubisoft, Playtika, Gameloft, Wargaming та Plarium. Вони надають тисячі робочих місць українським розробникам, при цьому працюючи за світовими стандартами. Існує низка студій, які були створені в країні і базуються локально, як Frogwares, Room 8 Studio, GSC Game World і 4A Games, проте кількість робітників в них значно менша. Локальний виробник страждає через недостатній розвиток IT в країні, через що іноземні видавці та групи студій мають більші представництва тут, відносно тих, які були засновані тут.

Через відносно невеликий рівень достатку населення, в країні широко розвивається тимчасова робота на замовлення, яка має назву аутсорс, тобто робота з “зовнішнього джерела”. Такою назвою іменуються будь які типи робіт,

які є додатковими та на які у замовника піде більша кількість ресурсів для реалізації, які реалізуються віддалено, на потужностях інших команд.

Одним з факторів, який відрізняє процес ситуацію з рішеннями для розробки будь-якого продукту в сфері комп'ютерної графіки та програмування є велика кількість безкоштовного програмного забезпечення, який не потребує ліцензування та часто розробляється з відкритим набором коду та інших складових. Це дає змогу зекономити на дорогому софті, якого не існувало до розквіту системи розповсюдження коду, як GitHub.

У перспективі, якщо буде створена деяка підприємницька ніша в напрямі, це матиме позитивні наслідки для фінансової екосистеми України. в результаті розвитку ігрової індустрії в Україні буде створена велика кількість робочих місць в сфері інформаційних технологій, що призведе до загального підвищення технологічного рівня країни. Крім того, існує низка професій, пов'язаних майже виключно з виробництвом та дизайном ігор, такі як дизайнери рівнів, ігрові композитори, художники спрайтів (двовимірних зображень, які використовують для анімацій), представники яких зможуть працювати на місцевого виробника. Результатом буде створена ніша для професій, в яких у експертів в цих галузях буде можливість реалізувати себе локально.

Беручи до уваги той факт, що ринок э глобальним, можна зробити висновок, що представництво нашої сцени на ньому допоможе залучити іноземний капітал в економіку нашої країни та привабити експертів з сусідніх країн. Це в свою чергу може стати стабільною та досить прибутковою частиною економіки України.

ВИСНОВОК ДО РОЗДІЛУ 1

1. Ігри – це доволі старий вид медіа, який тісно пов’язаний з історією людства, починаючи ще з витоків цивілізації. Це означає, що високий попит на такий вид розваг був завжди, і скоріш за все буде в майбутньому.
2. Ігрова індустрія в Україні має великий потенціал через малу кількість конкурентів на локальному ринку і доступ до світового ринку, тож і залучення іноземного капіталу та гру на світовому ринку. Вона може віднайти велику кількість інвестицій для країни у майбутньому.
3. На сьогоднішній день існує велика база програмного забезпечення та знань, яка пов’язана з принципами дизайну та розробки ігрових продуктів. Це означає, що завжди є доступ навіть для рішень, які були пропрієтарними, або потребували великої потужності на початку розвитку індустрії.
4. Ігрова індустрія – це ринок, який швидко розвивається та має позитивну динаміку росту. Тенденції показують, що вона і надалі буде прибутковою галуззю.
5. Ігри – це медіа. Незважаючи на те, що користувач теоретично може грати в одну гру весь час, не купуючи нових продуктів, як із будь-яким представником медійного продукту, система жанрів заохочує до того, щоб пробувати нові продукти, використовуючи елементи з тих, з якими користувач вже знайомий. Це створює екосистему, у якій схожі ігри не тільки конкурують за спільну аудиторію, а й навчають гравця принципам та специфіці певних жанрів та піджанрів, зменшуючи поріг входу для користування продукту.
6. Підбиваючи підсумки, в результаті дослідження була доведена доцільність розробки продукту та його цінність.

РОЗДІЛ 2. ДОСЛІДЖЕННЯ МЕТОДІВ РОЗРОБКИ ІГРОВИХ ПРОДУКТІВ. РОЗПОВСЮДЖЕНІ ФРЕЙМВОРКИ ТА ПРАКТИКИ У СФЕРІ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

2.1 Управління проєктами в сфері IT

Проєкт - тимчасовий процес, який має чітко визначений час початку та завершення, набір завдань і бюджет, розроблений для досягнення чітко визначеної мети чи завдання. Також існує визначення для менш визначених та жорстких рамок, що енкомпосує фреймворки з плавкими графіками і гнучкими цілями: група взаємопов’язаних заходів, обмежених часом, вартістю та обсягом, призначених для досягнення унікальної мети

Не всі проекти однакові, і вони відрізняються рядом різних елементів, які роблять кожен проект індивідуально унікальним. Ці фактори відрізняються в різних проектах, і їх необхідно враховувати, щоб проектами можна було ефективно й результативно керувати щодо кожного типу проекту через варіації їх розмірів і структури. Серед характеристик проєкту виділяють такі:

- **Обсяг проекту.** Він описує охоплення та масштаб проекту; це задає тон успіху проекту. Крім того, він визначає деталі роботи, яка має бути виконана в проекті. Обсяг проекту залежить від кількості залучених людей і масштабу впливу його результатів. Проекти можуть бути великими або малими залежно від обсягу.
- **Часові рамки.** Часові рамки проекту визначаються від його ініціації або концепції до оцінки результатів. Це період, протягом якого очікується завершення проекту. Часові рамки проекту також можна розділити на менші блоки, які самі по собі мають власний часовий проміжок.
- **Організація.** Ключова характеристика ефективного планування проекту. Організація проекту стосується того, як організовані завдання та дії та як пріоритетні типи проектів. Робочий процес проекту розраховується в кожному проекті для досягнення бажаних цілей. Інструмент управління та

планування проектів використовує такі технології, як PERT і CPM, щоб розрахувати робочий процес кожного проекту та знайти його найбільш оптимізований робочий шлях, а також різні типи інструментів управління проектом, щоб полегшити процес виконання проекту.

- **Вартість.** Проекти можуть бути дорогими або відносно дешевими залежно від їх загальної вартості. Додатки як Jira дозволяють планувати витрати на ваш проект разом із будь-якими оновленнями вартості, введеними вашою командою, у режимі реального часу, таким чином дозволяючи вам не витрачати кошти.
- **Комунікація.** Комунікація є наріжним каменем кожного проекту. Серед різних типів проектів спілкування, його частота та формат можуть відрізнятися. Однак без ефективної комунікації проект зазнає краху. Ефективна комунікація є важливим інструментом, який сприяє успіху будь-якого проекту, незалежно від його розміру. Велика кількість сучаного забезпечення дозволяє оптимізувати спілкування за допомогою онлайн-чату в реальному часі між членами команди та керівниками проектів, створюючи тим самим правильну взаємодію між ними.
- **Типи управління зацікавленими сторонами проекту.** Проекти можуть відрізнятися залежно від кількості залучених зацікавлених сторін. Іноді єдиними зацікавленими сторонами, які беруть участь у проекті, є команда та керівник проекту, але найчастіше це ширша група зацікавлених сторін. Чим більше зацікавлених сторін, тим складнішим є управління їхніми очікуваннями та спілкування. Менеджер може вибирати між цими типами лідерства.
- **Призначення завдань**. У різних типах проектів в управлінні проектами існує багато різних завдань і заходів. Проекти можуть відрізнятися залежно від того, як ці завдання розподіляються між членами команди – чи будуть вони виконуватися окремими членами чи групами та як буде визначено обов’язки. Керівник проекту повинен переконатися, що завдання проекту належним чином розподіляються між потрібними членами команди.

2.2 Розповсюджені фреймворки в IT

Управління проектом — це систематичне планування, організація та контроль виділених ресурсів для досягнення цілей і результатів проекту. Фреймворк управління проектом — це набір стандартних процесів управління проектом, шаблонів і інструментів, які можна використовувати для ініціювання, планування, виконання, контролю та закриття проекту. Наявність такої структури полегшує прийняття рішень, комунікацію та координацію між усіма проектами в портфоліо та, у свою чергу, сприяє чіткості управління та менеджменту. Зрештою, це призводить до більш ефективного використання корпоративних ресурсів.

Є певні критерії, які визначають проект. Проект визначається як діяльність або серія заходів із визначеним початком і кінцем. Проект має давати визначені результати, реалізовувати конкретні результати та підтримувати цілі державної політики, все в межах чіткого графіка та плану ресурсів. Управління проектом має здійснюватися в межах обсягу, часу, вартості та параметрів ефективності. Серед розповсюджених:

- **Waterfall.** Waterfall забезпечує посилений контроль на кожному етапі, але може бути дуже негнучким, якщо обсяг проєкту змінюється після того, як він уже запущений. Він пропонує більш формальний етап планування, який може збільшити шанси зафіксувати всі вимоги проєкту наперед, зменшуючи втрату будь-якої ключової інформації та вимог на початкових етапах.
- **Методологія PMI/PMBOK.** Методологія PMI/PMBOK охоплює розподіл різних типів проєктів на п’ять проєктних груп, погоджених Інститутом управління проєктами (PMI). По суті, це стосується життєвого циклу управління проєктом і вимог кожного етапу.
- **Agile.** Фреймворк Agile було розроблено в 2001 році. Він зосереджений на ефективній реакції на зміни, комплексній документації та взаємодії людей

через процеси та інструменти. Безперервна співпраця є ключовою особливістю як між членами команди, так і іншими зацікавленими сторонами проєкту.

- **Scrum**. Scrum є одним із різновидом сімейства методологій Agile і є її найпопулярнішою структурою. Він простий у реалізації та вирішує багато проблем, з якими стикаються розробники програмного забезпечення, наприклад заплутані цикли розробки, затримки у виробництві та негнучкі плани проєкту. Невелику команду зазвичай очолює так званий Scrum Master, який усуває всі перешкоди, що заважають ефективній роботі. Команди працюють у «спринтах», які є короткими циклами, які зазвичай складаються з двох тижнів і зазвичай збираються щодня, щоб обговорити хід виконання завдань свого проєкту.
- **Канбан.** Канбан – це методологія, заснована на здатності команди виконувати роботу. Він походить від Toyota в 1940-х роках. Це візуальний підхід до управління проєктами, який корисний для роботи, яка потребує постійного результату. Тут команди візуально переміщуються по ходу свого проєкту, що дозволяє чіткіше визначити будь-які перешкоди або вузькі місця, які можуть виникнути на цьому шляху.

2.1.1. Waterfall

Фреймворк “waterfall”, або так званий метод водоспаду, це один із найпростіших підходів для розробки, який застосовується для виконання широкого спектру проєктів. Протягом багатьох років водоспад був основною методологією управління проєктами. Він послідовний за своєю природою і використовується в багатьох галузях, найчастіше в розробці програмного забезпечення. Він містить статичні фази, тобто аналіз вимог, проєктування, тестування, впровадження та обслуговування, які виконуються у попередньо визначеному порядку.

Послідовні фази в моделі водоспаду представлені етапами:

- **Збір і аналіз вимог**. Усі можливі вимоги до системи, яка буде розроблена, фіксуються на цьому етапі та документуються в документі специфікації вимог.
- **Проектування системи.** Специфікації вимог з першого етапу вивчаються на цьому етапі та готується проект системи. Цей дизайн системи допомагає визначити апаратні та системні вимоги, а також допомагає визначити загальну архітектуру системи.
- **Впровадження.** За допомогою вхідних даних проектування системи система спочатку розробляється в невеликих програмах, які називаються модулями, які інтегруються на наступному етапі. Кожен блок розробляється та перевіряється на його функціональність, що називається модульним тестуванням.
- **Інтеграція та тестування.** Усі блоки, розроблені на етапі впровадження, інтегруються в систему після тестування кожного блоку. Після інтеграції вся система перевіряється на будь-які помилки та збої.
- **Розгортання системи.** Після завершення функціонального та нефункціонального тестування; продукт розгортається в середовищі клієнта або випускається на ринок.
- **Технічне обслуговування.** У клієнтському середовищі виникають деякі проблеми. Для вирішення цих проблем випускаються патчі – додаткові оновлення з виправленнями. Крім того, для вдосконалення продукту випущено кілька кращих версій. Технічне обслуговування виконується для внесення цих змін у клієнтське середовище.

Усі ці фази розташовуються каскадом одна до одної, у якій прогрес розглядається як постійний рух вниз (як водоспад) через фази. Наступна фаза починається тільки після того, як визначений набір цілей досягнуто для попередньої фази, і саме тому методологія і отримала таку назву. У цій моделі фази не накладаються.

Кожне розроблене програмне забезпечення є різним і вимагає відповідного підходу, який слід застосовувати на основі внутрішніх і зовнішніх факторів. Деякі ситуації, коли використання моделі waterfall є найбільш прийнятним, це:

- Вимоги дуже добре задокументовані, чіткі та фіксовані.
- Визначення продукту стабільне.
- Технологія зрозуміла і не є динамічною.
- Немає неоднозначних вимог.
- Доступні достатні ресурси з необхідним досвідом для підтримки продукту.
- Проект є коротким.

Переваги водоспадного розвитку полягають у тому, що він дозволяє розділяти та контролювати. Можна встановити графік із кінцевими термінами для кожного етапу розробки, і продукт може проходити через етапи моделі процесу розробки одну за одною. Розробка проходить від концепції до проектування, реалізації, тестування, інсталяції, усунення несправностей і закінчується експлуатацією та обслуговуванням[7]. Кожна фаза розвитку протікає в строгому порядку.

Основні переваги моделі водоспаду:

- Простий і легкий для розуміння та використання
- Легко керувати завдяки жорсткості моделі. Кожен етап має конкретні результати та процес перевірки.
- Фази обробляються та завершуються одна за одною.
- Добре працює для невеликих проектів, де вимоги добре зрозумілі.
- Чітко визначені етапи.
- Добре зрозумілі віхи.
- Легко розставляти завдання.
- Процес і результати добре задокументовані.

Основні недоліки моделі водоспаду:

- Жодне робоче програмне забезпечення не створюється до кінця життєвого циклу.

- Високий рівень ризику та невизначеності.
- Не дуже хороша модель для складних і об’єктно-орієнтованих проектів.
- Погана модель для тривалих і поточних проектів.
- Не підходить для проектів, де вимоги мають помірний або високий ризик зміни. Таким чином, ризик і невизначеність є високими з цією моделлю процесу.
- Важко виміряти прогрес в рамках етапів.
- Неможливо задовольнити мінливі вимоги.
- Коригування обсягу протягом життєвого циклу може завершити проект.
- Інтеграція здійснюється як «великий вибух» у самому кінці, що не дозволяє завчасно виявити будь-які технологічні чи бізнес-вузькі місця чи проблеми.

Побудована за таким методом, діаграма проєкту виглядатиме як діагональна лінія з різними послідовними роботами (рис. 2.1.).

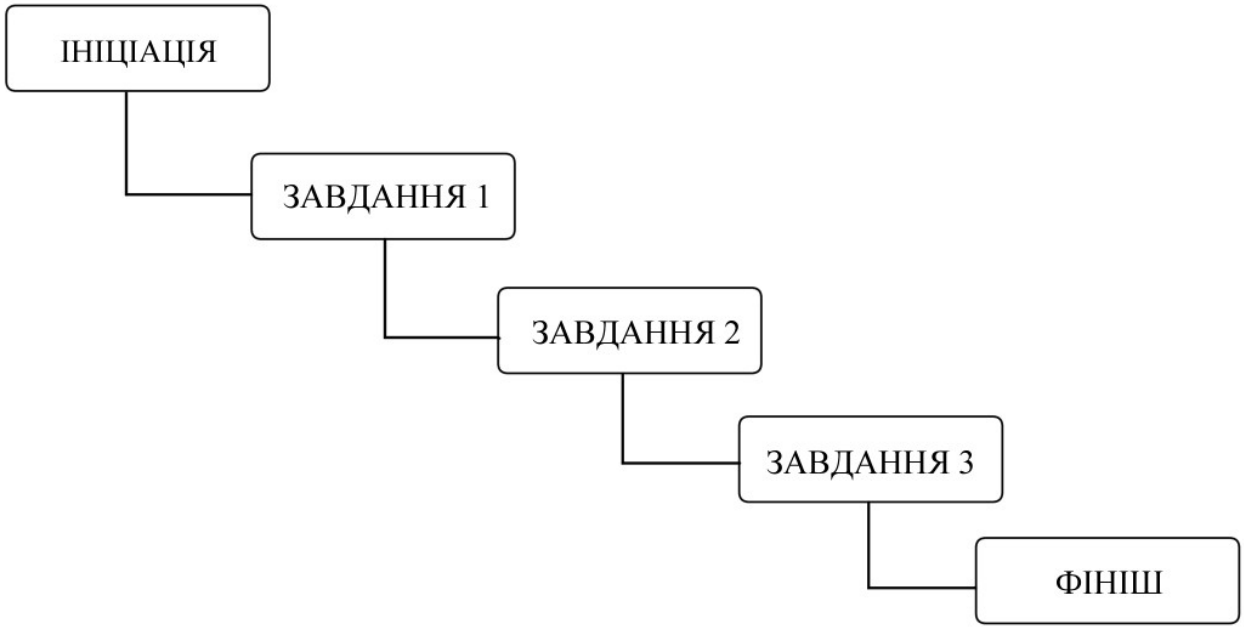


Рис. 2.1. Схема проєкту, побудованого за методом водоспаду

2.1.2. V-модель

Логічним продовженням розвитку моделі водоспаду є так звана V-модель. V-модель є розширенням моделі водоспаду та базується на асоціації фази тестування для кожної відповідної стадії розробки. Це означає, що для кожної окремої фази циклу розробки існує безпосередньо пов'язана фаза тестування. Це дуже строга модель, і наступний етап починається лише після завершення попереднього.

У V-моделі є кілька етапів перевірки, кожен із яких детально пояснюється нижче[8]:

- **Аналіз бізнес-вимог.** Це перша фаза циклу розробки, на якій вимоги до продукту розуміються з точки зору клієнта. Цей етап передбачає детальне спілкування з клієнтом, щоб зрозуміти його очікування та точні вимоги. Це дуже важлива діяльність, і нею потрібно добре керувати, оскільки більшість клієнтів не впевнені, що саме їм потрібно. На цій стадії здійснюється планування проекту приймального випробування, оскільки бізнес-вимоги можна використовувати як вхідні дані для приймального випробування.
- **Проектування системи.** Коли у вас є чіткі та детальні вимоги до продукту, настав час розробити повну систему. Проект системи містить розуміння та деталізацію повного обладнання та налаштування зв'язку для продукту, що розробляється. План тестування системи розробляється на основі проекту системи. Виконання цього на ранньому етапі залишає більше часу для фактичного виконання тесту пізніше.
- **Архітектурне проектування.** Архітектурні характеристики розуміються та проектуються на цьому етапі. Зазвичай пропонується більше ніж один технічний підхід, і на основі технічної та фінансової здійсненності приймається остаточне рішення. Конструкція системи далі розбивається на модулі, які виконують різні функції. Це також називається дизайном високого рівня (HLD). Передача даних і зв'язок між внутрішніми модулями та із зовнішнім світом (іншими системами) чітко зрозумілі та визначені на цьому етапі. З цією інформацією можна розробити та задокументувати інтеграційні тести на цьому етапі.

- **Дизайн модуля.** На цьому етапі вказується детальна внутрішня конструкція для всіх системних модулів, що називається низькорівневим проектом (LLD). Важливо, щоб дизайн був сумісний з іншими модулями архітектури системи та іншими зовнішніми системами. Модульні тести є важливою частиною будь-якого процесу розробки та допомагають усунути максимальну кількість недоліків і помилок на дуже ранній стадії. Ці модульні тести можуть бути розроблені на цьому етапі на основі дизайну внутрішнього модуля.
- **Фаза кодування.** Фактичне кодування системних модулів, розроблених на етапі проектування, виконується на етапі кодування. Найкраща підходяща мова програмування визначається на основі системних і архітектурних вимог. Кодування виконується на основі рекомендацій і стандартів кодування. Код проходить численні перевірки коду та оптимізується для найкращої продуктивності перед тим, як остаточна збірка буде перевірена в сховищі.
- **Модульне тестування.** Модульні тести, розроблені на етапі проектування модуля, виконуються на коді під час цього етапу перевірки. Модульне тестування — це тестування на рівні коду, яке допомагає усунути помилки на ранній стадії, хоча модульне тестування не може виявити всі дефекти.
- **Інтеграційне тестування.** Інтеграційне тестування пов'язане з етапом архітектурного проектування. Інтеграційні тести виконуються для перевірки співіснування та зв’язку внутрішніх модулів у системі.
- **Тестування системи.** Тестування системи безпосередньо пов'язане з етапом проектування системи. Системні тести перевіряють всю функціональність системи та зв'язок системи, що розробляється, із зовнішніми системами. Більшість проблем із сумісністю програмного та апаратного забезпечення можна виявити під час виконання цього тесту системи.
- **Приймальні випробування.** Приймальні випробування пов’язані з фазою аналізу бізнес-вимог і включають тестування продукту в середовищі користувача. Приймальні тести виявляють проблеми сумісності з іншими

системами, доступними в середовищі користувача. Він також виявляє нефункціональні проблеми, такі як дефекти навантаження та продуктивності в реальному середовищі користувача.

Застосування V-моделі майже таке ж, як і моделі водоспаду, оскільки обидві моделі мають послідовний тип. Вимоги мають бути дуже чіткими перед початком проекту, оскільки повертатися назад і вносити зміни зазвичай дорого. Ця модель використовується в галузі медичних розробок, оскільки це суворо дисциплінована область[9].

- Вимоги чітко визначені, чітко задокументовані та фіксовані.
- Продукту є чітко визначеним.
- Технологія не є динамічною і добре зрозуміла команді проекту.
- Немає двозначних або невизначених вимог.
- Проект є коротким.

Перевага методу V-моделі полягає в тому, що його дуже легко зрозуміти та застосувати. Простота цієї моделі також полегшує керування. Недоліком є те, що модель не гнучка до змін, і лише у випадку зміни вимог, що дуже часто зустрічається в сучасному динамічному світі, внесення змін стає дуже дорогим.

Переваги методу V-моделі:

- Це високодисциплінована модель, і фази завершуються по черзі.
- Добре працює для невеликих проектів, де вимоги добре зрозумілі.
- Простий і легкий для розуміння та використання.
- Легко керувати завдяки жорсткості моделі. Кожен етап має конкретні результати та процес перевірки.

Недоліки методу V-моделі:

- Високий ризик і невизначеність.
- Не дуже хороша модель для складних і об’єктно-орієнтованих проектів.
- Погана модель для тривалих і поточних проектів.
- Не підходить для проектів, де вимоги мають помірний або високий ризик зміни.

- Коли програма знаходиться на стадії тестування, важко повернутися назад і змінити функціональність.
- Жодне робоче програмне забезпечення не створюється до кінця життєвого циклу.

Рис. 2.2. Схема проєкту, побудованого за V-методом

2.1.3. Сімейство фреймворків Agile

Agile використовує значно відмінний підхід до управління проектами. Спочатку він був розроблений для проектів, які вимагають значної гнучкості та швидкості. Щоб досягти цього, Agile складається з коротких циклів доставки, тобто «спринтів». Agile може найкраще підходити для проектів, які вимагають менше контролю та спілкування в режимі реального часу в рамках самомотивованої команди та зацікавлених осіб.

Незважаючи на популярну думку, Agile не є методологією, а являє собою набір настанов, принципів і механік, на плацдармі яких повинні вибудовуватися

методології, які були зафіксовані в так званому “маніфесті Agile”, який був розроблений експертами галузі IT[10].

Ці принципи включають в себе 12 наступних пунктів:

1. Задоволення клієнтів завдяки ранній і постійній доставці цінного програмного забезпечення.
2. Вітати зміну вимог, навіть на пізній стадії розробки.
3. Доставляйте робоче програмне забезпечення часто (тижнями, а не місяцями).
4. Тісна щоденна співпраця між бізнесменами та забудовниками.
5. Проекти будуються навколо мотивованих людей, яким варто довіряти.
6. Розмова віч-на-віч є найкращою формою спілкування (co-location).
7. Працююче програмне забезпечення є основним показником прогресу.
8. Сталий розвиток, здатний підтримувати постійний темп.
9. Постійна увага до технічної досконалості та гарного дизайну.
10. Простота — мистецтво максимізації обсягу невиконаної роботи — є важливою.
11. Найкращі архітектури, вимоги та проекти виходять із самоорганізованих команд.
12. Команда регулярно розмірковує над тим, як стати ефективнішими, і відповідно коригується.

Agile базується на адаптивних методах розробки програмного забезпечення, тоді як традиційні моделі SDLC, такі як модель водоспаду, базуються на прогнозному підході. Групи прогнозування в традиційних моделях SDLC зазвичай працюють з детальним плануванням і мають повний прогноз точних завдань і функцій, які будуть доставлені в найближчі кілька місяців або протягом життєвого циклу продукту[11].

Методи прогнозування повністю залежать від аналізу потреб і планування, зробленого на початку циклу. Будь-які зміни, які необхідно внести, проходять суворий контроль за змінами та встановлення пріоритетів.

Рис. 2.2. Цикл розробки Agile

Agile використовує адаптивний підхід, де немає детального планування та є ясність щодо майбутніх завдань лише стосовно того, які функції потрібно розробити. Існує розробка, керована функціями, і команда динамічно адаптується до мінливих вимог до продукту. Продукт тестується дуже часто через ітерації випуску, що мінімізує ризик будь-яких серйозних збоїв у майбутньому.

Гнучкі методи останнім часом широко поширені у світі програмного забезпечення. Однак цей метод не завжди підходить для всіх продуктів. Ось деякі переваги та недоліки моделі Agile.

Переваги гнучких принципів Agile:

- Це дуже реалістичний підхід до розробки програмного забезпечення.
- Сприяє командній роботі та перехресному навчанню.
- Функціональність можна швидко розвинути та продемонструвати.
- Вимоги до ресурсів мінімальні.
- Підходить для постійних або змінних вимог
- Надає ранні часткові робочі рішення.
- Хороша модель для середовища, яке постійно змінюється.
- Мінімум правил, документацію легко використовувати.
- Планування не потрібне або мало потрібно.

- Легко керувати.
- Надає гнучкість розробникам.

Недоліки гнучких принципів Agile:

- Не підходить для обробки складних залежностей.
- Більший ризик сталості, ремонтопридатності та розширюваності.
- Загальний план, гнучкий лідер і гнучка практика PM є обов'язковими, без яких це не працюватиме.
- Суворе управління доставкою визначає обсяг, функціональні можливості, які необхідно надати, і коригування для дотримання термінів.
- Значною мірою залежить від взаємодії з клієнтом, тому, якщо клієнт не зрозумілий, команду можна направити в неправильному напрямку.
- Існує дуже висока індивідуальна залежність, оскільки створюється мінімум документації.
- Передача технології новим членам команди може бути досить складною через відсутність документації.

Як фреймворк управління проектами, Agile є дуже інтерактивним, що дозволяє швидко коригувати весь проект. Він зазвичай використовується в проектах розробки програмного забезпечення значною мірою тому, що полегшує швидке виявлення проблем і внесення змін на ранній стадії процесу розробки, замість того, щоб чекати завершення тестування. Agile пропонує повторювані процеси, знижує ризики, забезпечує миттєвий зворотний зв'язок, забезпечує швидкий оборот і зменшує складність.

2.1.4.Kanban

Kanban — це система контролю ресурсами, яка використовується у виробництві «точно вчасно» (JIT). Він був розроблений Тайічі Оно, промисловим інженером Toyota, і отримав свою назву від кольорових карток, які відстежують виробництво та замовляють нові поставки деталей або матеріалів, коли вони

закінчуються. Канбан — це японське слово, яке безпосередньо перекладається як «нотатка», тому система канбан просто означає використання візуальних підказок для спонукання до дії, необхідної для підтримки процесу.

Метод має кілька основних принципів, які визначають, як відбуваються процеси та як члени команди мають бути залучені до процесу. В основі канбану процес має бути візуально зображений. Незалежно від того, чи то за допомогою фізичних, матеріальних карток чи за допомогою технології та програмного забезпечення, процес має бути показано крок за кроком за допомогою візуальних підказок, які роблять кожне завдання чітким. Ідея полягає в тому, щоб чітко показати, що таке кожен крок, які очікування та хто виконуватиме які завдання.

До старомодних (але використовуваних досі) методів входило складання завдань канбану на нотатках. Кожна липка нотатка може мати різний колір, щоб позначити різні типи робочих елементів. Потім ці завдання будуть розміщені в доріжках для плавання, визначених розділах, які групують пов’язані завдання для створення більш організованого проекту. Сьогодні програмне забезпечення для управління запасами зазвичай керує процесом канбан.

Оскільки канбан базується на ефективності, мета канбану — мінімізувати обсяг незавершеної роботи. Командам пропонується виконати попередні завдання, перш ніж переходити до нового. Це гарантує, що майбутні залежності можна буде розпочати раніше, а ресурси, наприклад персонал, не будуть неефективно чекати, щоб розпочати виконання завдання, покладаючись на інших.

Під час виконання процесу канбан компанія повинна внутрішньо оцінити відповідну кількість WIP, яку потрібно мати. Це часто пов’язано з кількістю людей у процесі; зі зменшенням кількості працівників, прив’язаних до проекту, зменшується і дозволена кількість елементів, над якими працюється. Це обмеження також повідомляє іншим командам або відділам, що вони повинні бути уважними до своїх запитів до інших команд, оскільки на кожну групу осіб можуть бути накладені робочі обмеження.

Під час виконання процесу компанія зможе визначити сильні та слабкі сторони в процесі роботи. Іноді обмеження не дотримуються або цілі не

досягаються; у цьому випадку команда повинна керувати робочим процесом і краще розуміти недоліки, які необхідно подолати.

Важливою частиною канбану є спостереження та усунення вузьких місць до їх появи. Це включає прогнозування виробництва та використання ресурсів. Коли процес стає більш передбачуваним, компанії стає легше брати на себе зобов'язання перед клієнтами або робити процеси ще ефективнішими, повністю зменшуючи додаткові невикористані ресурси.

У рамках візуального зображення робочих процесів процеси часто чітко визначені. Відділи часто можуть легко зрозуміти очікування, які покладаються на їхні команди, а картки канбан, призначені конкретним особам, чітко визначають обов'язки для кожного завдання. Чітко визначивши політику, кожен працівник зрозуміє, що від нього очікується, які критерії контрольного списку мають бути виконані перед завершенням і що відбувається під час переходу між кроками.

Рис. 2.3. Дошка Kanban

Використовуючи метод канбану, компанії часто збирають інформацію, аналізують, як протікає процес, і вносять зміни для подальшого вдосконалення процесу. Цей цикл зворотного зв'язку дозволяє співробітникам постійно вдосконалюватися та робити поступові невеликі вдосконалення, до яких легше адаптуватися. Відгук може бути позитивним або негативним. Підхід канбан

полягає в розумінні невдач на ранніх стадіях процесу та швидкому їх усуненні; це дозволяє компанії адаптуватися до правильного шляху до того, як неефективність стане більшою проблемою.

Оскільки завдання розбиті на дуже маленькі картки канбану, люди часто повинні покладатися один на одного, використовуючи метод. Люди, часто в різних командах, повинні співпрацювати та обговорювати переходи між плавальними доріжками, тоді як інші люди повинні об'єднуватися в групи, щоб швидко виявляти та вирішувати проблеми. Відповідно до канбану, зміни в процесі повинні бути широко повідомлені, оскільки зміни, внесені в одній сфері, можуть мати більший вплив в іншій.

2.1.5.Scrum. Scrum проти Kanban

Scrum — це структура управління проектами, яка забезпечує структуру для швидкозмінних команд Agile для визначення пріоритетів, керування та виконання роботи. Ця структура допомагає командам надавати цінність клієнтам, часто та ефективно співпрацювати та вирішувати нагальні проблеми розвитку. Scrum також полегшує процес доставки продукту.

Церемонії Scrum, артефакти та ролі забезпечують структурований підхід до гнучкого управління проектами. Кожна зустріч, роль і артефакт Scrum діють як частина головоломки, що дозволяє командам розставляти пріоритети, безперервно виконувати завдання та ефективно вирішувати проблеми.

Через це розробка продуктів Scrum є досить популярною. Згідно зі звітом Інституту управління проектами, 55% менеджерів проектів підтвердили, що вони використовують Scrum принаймні час від часу[12].

Принципи Scrum служать керівництвом для впровадження методології Scrum. Вони забезпечують розуміння процесів і динаміки середовища Scrum. Обов'язком Scrum-майстра є дотримання принципів і цінностей Scrum. Нижче приведені шість принципів Scrum:

- **Емпіричне управління процесом.** Щоб залишатися ефективним, гнучким і здатним реагувати на зміни, Scrum покладається на емпіричний контроль процесів. Це означає, що весь процес Scrum керується прозорістю, перевіркою та адаптацією.
- **Самоорганізація.** Метод Scrum заохочує певний рівень незалежності від команди Scrum. Коли Scrum-команди називають «самоорганізованими», це просто означає, що вони самостійно керують своїми завданнями, самостійно вирішують проблеми та відповідають перед собою та один перед одним, а не перед зовнішнім менеджером.
- **Таймбокс.** Таймбокс – це практика, коли фіксований проміжок часу виділяється на певні дії чи цілі. Розмежування часу дозволяє виконувати дії в оптимальні часові рамки, не триваючи надто довго. Цей процес ідеально підходить для встановлення часових рамок для таких заходів, як планування спринту та ретроспектива спринту.
- **Розстановка пріоритетів на основі цінностей.** Щоб отримати пріоритети на основі цінностей, елементи в резерві продукту постійно оновлюються на основі їхньої цінності та важливості для кінцевого користувача та зацікавлених сторін.
- **Ітераційна розробка.** Через постійні спринти цілі розробки продукту постійно переглядаються та оновлюються, щоб створити продукт найкращої якості та процес доставки.
- **Співпраця.** Scrum-команди співпрацюють часто і, іноді, довго. Щоденні стендап-зустрічі — це можливість співпрацювати та вирішувати проблеми, як і спринт-перегляди та ретроспективи.

І Scrum, і Kanban містять методології, які допомагають компаніям працювати ефективніше. Однак кожен має дуже різні підходи до досягнення такої ефективності. Scrum-підходи встановлюють певні часові рамки для внесення змін; в ці періоди відбуваються специфічні зміни. За допомогою канбану зміни вносяться постійно.

Методологія Scrum розбиває завдання на спринти, визначені періоди з початковим і кінцевим періодами, у яких завдання чітко визначені та мають бути виконані певним чином. Жодних змін чи відхилень від цих часових рамок або завдань не повинно відбуватися. Scrum часто вимірюється швидкістю або запланованою потужністю, а власник продукту або “Scrum master” – менеджер, відповідальний за дотримання концепцій Scrum – контролює процес[13].

З іншого боку, канбан більш адаптивний, оскільки він аналізує те, що було зроблено в минулому, і вносить постійні зміни. Команди встановлюють власну частоту або цикли, і ці цикли часто змінюються за потреби. Kanban вимірює успіх, вимірюючи тривалість циклу, пропускну здатність і незавершену роботу.

Після порівняння цих двох підходів було вирішено використати Scrum для виконання проєкту. Він є більш чітким та визначеним у часі відносно Kanban, та переваги другого через адаптацію до змін не є ключовим фактором для поставленого завдання. Так як буде розроблений план роботи з усіма роботами, які повинна виконати команда, щоб досягнути результату, тобто завершеного продукту, був вибраний більш поміркований і строгий метод.

2.2. Звітна документація. Типи звітної документації

Звіти є центральною частиною управління проєктами. Вони важливі для інформування відповідних департаментів та стейкхолдерів уцілому про поточний стан проєкту. Звіти надають можливість подивитися на обстановку ходу робіт в переспективі[14].

Ось кілька прикладів деяких різних типів звітів про проєкт, які мають заповнювати керівники проєктів:

- **Звіти про стан виконання**. Звіти про стан зазвичай складаються для надання спонсорам або певним зацікавленим сторонам, на яких вплинуть результати проєкту, і повинні постійно оновлюватись щодо прогресу проєкту. Це найпоширеніший тип звіту, над яким менеджери проєктів працюють регулярно. Вони можуть бути щотижня або щомісяця, і їх

частота залежатиме від стадії, на якій перебуває ваш проєкт, і від того, скільки є інформації для докладу.

- **Звіти про ризики.** Аналіз та управління ризиками є ключовими параметрами управління проєктами. Ризики необхідно аналізувати та тримати під контролем, щоб не завдати серйозної шкоди проєкту. Про ризики повідомляють щонайменше щомісяця, і зазвичай вони висвітлюються після наради з розгляду ризиків. Цей звіт містить профіль ризиків проєкту та те, як проєктний менеджер керує або збираєтеся керувати ризиками.
- **Звіти управління.** Кожна доповідь має бути написана з урахуванням аудиторії, яку вона охопить. Для звітів ради проєкту слід підтримувати високий рівень деталізації проєкту. Різні типи звітів про організацію проєкту мають різні формати документів, але всі вони завжди дуже детальні. Це може допомогти в тому сенсі, що члени ради можуть точно визначити певні проблемні області, з якими вони можуть допомогти: чим більше знань про проблемну область, тим більше шансів на успіх. Він має бути у форматі, який легко читати та якого легко дотримуватися.
- **Ресурсні звіти.** Розробка розподілу ресурсів вручну займе багато часу, особливо якщо проєкт великомасштабний і більш складний. Програмне забезпечення для керування проєктами, таке як Sinnaps, використовує модернізовану версію діаграми. По суті, звіт про ресурси сам по собі допомагає визначити, хто що робить з якими ресурсами та протягом якого періоду часу. Це допомагає гарантувати відсутність конфліктів ресурсів і робить проєкт ефективнішим у цілому. Цей тип звіту, мабуть, є одним із найкорисніших для керівників.

Для даного проєкту будуть використані декілька видів звітів: ресурсні та звіти статусу виконання. Для того, щоб менеджер міг тверезо оцінювати ситуацію та коригувати курс розробки доволі плавкого в плані виробництва продукту потрібно адаптувати графіки відносно витрат. Чим вище час створення рішення,

тим більше ентропії генерує проєкт, і єдине, що може попередити додаткові витрати – це грамотний менеджмент та розподілення обов'язків і ресурсів.

2.3. Причини виникнення ризиків. Методи їх усунення

У кінці попереднього параграфу було згадано про те, що нестабільність в будь-якій системі збільшується з її комплексністю. Така властивість називається ентропією. У проєктах вона виражається в проблемах з комунікацією між відділами, ресурсами – трудовим та матеріальними, запланованими та спорадичними відставання та форс-мажорами[15].

Для того, щоб робота була виконана максимально чітко, швидко та з найменшим тертям, треба йти за повним планом чи планом на певний проміжок часу. Виконання такої роботи якраз і лягає на плечі проєктного менеджера.

Кожен спеціаліст визначає свої власні пріоритети для кожного окремого кейсу, так як самі проєкти можуть сильно відрізнятися в структурі, фінальному продукті, розміру команди та її досвіду. Далі категорії ризиків у проєктах:

1. Розповзання масштабу проєкту. Розповзання області відбувається, коли будь-який параметр проекту не був чітко визначений з самого початку або існує тиск, або всередині команди проекту, або ззовні, з боку клієнтів чи босів, щоб взятися за завдання, які не входили до початкового плану проекту.

Часто повзання масштабу є результатом великих намірів. У вашого клієнта є чудова нова ідея, або член команди мріє включити вражаючу функцію, або, можливо, хтось із вищого керівництва хоче зробити більше, щоб справити враження на клієнта. Незначні коригування тут і там можуть додати години (або дні чи тижні) додаткової роботи.

Проблема повзучого масштабу полягає в тому, що воно часто сприяє провалу проекту. Якщо керівники проєктів не передбачили в бюджеті час або ресурси, необхідні для виконання додаткових завдань, тому те, що могло б мати приголомшливий успіх, закінчується невдачею.

Як керувати ризиком: Щоб запобігти розповзанню обсягу, переконайтеся, що ваш клієнт точно знає, що йому потрібно, і включив усі деталі в письмовий запит. Створіть статутний документ проекту для свого клієнта, в якому описано, що та коли виконуватиме ваша команда, а також додайте розділ, який пояснює, що станеться, якщо клієнт доповнить узгоджені параметри.

Залежно від запиту, можливо, буде можливим пристосуватися до невеликого розширення обсягу, але обов'язково поясніть своєму клієнту, що такі доповнення призведуть до змін у вартості та/або часовому графіку проекту.

2. Розповзання бюджету Бюджетне розповзання тісно пов'язане з розповзанням обсягу проєкту. Зміни в масштабі проекту, безумовно, можуть негативно вплинути на прибуток, але також можуть негативно вплинути й інші фактори. Надто оптимістичні оцінки витрат можуть призвести до перевищення бюджету, наприклад, коли менеджер недооцінює час або зовнішні ресурси, необхідні для завершення проекту.

Іноді виникають непередбачені зміни матеріальних або трудових витрат. Погане планування майже завжди впливає на ваш бюджет, як і погана комунікація, і ми поговоримо про це нижче.

Як керувати ризиком: Деякі зміни бюджету знаходяться поза вашим контролем як керівника проекту, але не всі. Щоб пом'якшити ризики, пов'язані з бюджетом, дуже уважно проведіть дослідження та не представляйте остаточний бюджет, доки план і графік проекту не будуть завершені.

Створення прозорості навколо вашого проекту також може запобігти перевитратам бюджету; як клієнт, так і члени команди можуть допомогти зберегти проект у рамках бюджету, якщо вони мають доступ до відповідної інформації.

3. Проблеми комунікації. Як і в житті, комунікація є одним з найважливіших факторів успішного управління проектами, а погана комунікація становить величезний і непотрібний ризик. Виберіть хороші інструменти для спільної роботи над проектом і поясніть їх своїй команді на початку проекту.

Більшість команд використовують певну комбінацію електронної пошти, текстових повідомлень, служби чату та/або інтегрованих програм Google. Як керівник проекту, він повинен переконатися, що кожен член команди може використовувати вибрану вами технологію. Крім методу спілкування, переконайтеся, що чітко пояснюєте очікування щодо часу відповіді та подайте гарний приклад професійного стилю та тону спілкування.

Як керувати ризиком: щоб зменшити комунікаційні ризики, вам слід спростити комунікаційні потоки проекту до найменшої кількості, яка дозволить вашій команді залишатися ефективною. Часто це дослідження про те, які платформи для спілкування та співпраці найкраще прислужаться людям і проекту, яким менеджер керує. Вам також може знадобитися навчити деяких членів команди розвивати кращі навички спілкування.

4. Відсутність чіткості. Погана комунікація з боку клієнтів і зацікавлених сторін може створити ще один ризик: нечіткість вимог. Подібно до того, як вам потрібно налагодити хорошу комунікацію зі своєю командою, керівники також повинні розвивати хорошу комунікацію зі своїм клієнтом та іншими зацікавленими сторонами, щоб вимоги проекту були зрозумілими з самого початку. Більшість людей мали поганий досвід, коли витрачали багато часу на проект лише для того, щоб з’ясувати, що вони неправильно зрозуміли, про що їх запитували.

Як керувати ризиком: хороші навички слухання значною мірою зменшують цей ризик. Як керівник проекту, він повинен отримати роз’яснення від клієнта щодо того, що йому потрібно, і уважно вислухати всіх зацікавлених сторін проекту, коли вони надають свій внесок. Потрібно задавати стільки запитань, скільки потрібно, щоб отримати чітке уявлення про бажаний кінцевий продукт і його призначення.

Вимагайте чітких даних у формі чисел, але також запитуйте історії про те, як виглядає успіх для кінцевого користувача. Чітке спільне бачення може запобігти проблемам і стати джерелом натхнення для команди.

5. Погане планування. Планування проекту є головним компонентом успішного управління проектом, а погане планування може створити безліч ризиків для вашого проекту. Планування передбачає створення документа, сьогодні зазвичай цифрового документа, який детально описує графік проекту та організаційні ресурси, необхідні для виконання кожного завдання.

Розклад проекту має бути доступним кожному члену команди. Його мета полягає в тому, щоб передати критичну інформацію команді, тому він повинен бути вичерпним і легким для розуміння. Планування проєктів можна розбити на вісім кроків, і навіть якщо це може зайняти багато часу, належне програмне забезпечення для планування проекту може допомогти вам уникнути багатьох ризиків, які інакше можуть виникнути у вашому проекті.

Як керувати ризиком: детальне планування має важливе значення для створення розкладу проекту. Якщо керівник команди проєкту шивдко і добре планує, є багато інструментів планування проекту, які можуть виявитися корисними, щоб не відставати від плану. Залежно від обсягу та складності вашого проекту простий спільний календар може бути ефективним. Для більш складних проектів деякі команди використовують діаграми Ганта, а інші віддають перевагу дошкам Канбан.

2.4. Комунікація в команді. Програмні рішення для командної роботи

В сучасному світі спілкування між членами команди не повинно бути виключно очним. Великий спектр програм, які надають можливість дистанційного спілкування був розроблений за декілька останніх років. Найпопулярнішими рішеннями для онлайн-мітингів є Microsoft Teams, Cisco Webex і Zoom[16]. Це дозволяє працювати пліч-о-пліч з фахівцями, з якими неможливо працювати на локальному рівні через їх місце проживання чи перебування. Серед цих додатків істотно виділяється корпоративний месенджер Slack.

Абревіатура Slack означає “журнал усіх розмов і знань із можливістю пошуку”, і таку назву має доволі потужний продукт, використання якого наразі розповсюджене в сфері розробки програмного забезпечення[17]. Він принципово не відрізняється від своїх конкурентів в плані базового функціоналу. Проте головною відмінністю продукту є те, як структуровані канали спілкування. Slack представляє собою рішення, яке нагадує сучасні програми для спілкування як Teamspeak або Discord, які зарекомендували себе на ринку як зручні та прості для використання інструменти, тобто такі, які перенесені на лоно корпоративного програмного забезпечення допоможуть команді зменшити тертя під час виконання роботи.

Постає питання в доцільності використання підписного продукту для того, щоб ввести функцію віддаленого спілкування для команди, коли існують схожі рішення з такою ж структурою, як і у Slack. Головною причиною використання саме його є те, що цей додаток підтримує величезну кількість інтеграцій з іншим професійним програмним забезпеченням. Це означає, що для того, щоб наприклад створити нове завдання та видати його у зручній формі для спеціаліста, не потрібно використовувати спочатку один продукт для проєктування плану, другий для опису спринту, третій для дошки з завданнями і четвертий для безпосередньої передачі повідомлення про роботу, яку потрібно виконати.

Також, другим важливим програмним рішенням є програма менеджменту проєктами. У наш час основною програмою такого класу є Jira. Jira – це широко розповсюджений в практиці розробки програмного забезпечення додаток. На відміну від більш лінійного Microsoft Project, він дозволяє розробляти дошки з завданнями та спринтами у легкій та доступній формі. Визначною якістю продукту є те, що він дозволяє синхронізувати роботу різних відділів чи працівників, так як сам продукт реалізований у вигляді сайту, де кожен може зареєструватися та підключитися до вже існуючої команди, або створити свою. Програмне рішення було створене з урахуванням специфіки сучасного проєктного продуктового менеджменту та ідей Agile про безперервне покращення та виробництво нових продуктів[18].

Jira не потребує глибоких знань в своїй документації для того, щоб почати нею користуватися, що робить її корисним інструментом для організації робіт. Головною відмінністю програми є широкий функціонал, який надає їй лінійка іншого програмного забезпечення, яке належить розробнику Jira – Atlassian, – яке дозволяє отримати більше, ніж просто органайзер робіт.

Крім спілкування між членами команди їм знадобиться також деяке програмне рішення для синхронізації даних проєкту. Так як розробка буде вестися паралельно з візуальною складовою гри, для того щоб зекономити велику кількість часу, знадобиться залучити підтримку рішення для роботи з так званими репозиторіями файлів – певних ресурсів, які використовуватися команда під час безпосередньої роботи над проєктом. Найпопулярнішим рішенням для цього є сервіс GitHub.

Github – це безкоштовний менеджер репозиторіїв, який став штампом індустрії інформаційних технологій. Він надає можливості роботи одразу декількох ліній розробки в різних напрямках, управління версіями та колаборації. За допомогою цього продукту команда матиме постійний доступ до файлів проєкту та зможе працювати незалежно над різними частинами продукту, не сильно залежачи один від одного в плані затримок розробки функцій, за які відповідальні інші працівники[19].

Для реалізації цього проєкту потрібно буде налаштувати роботу між цими додатками та роботу між членами команди. У той час, коли розробник і дизайнери оформлення зможуть незалежно розроблювати різні складові продукту, тестувальники і менеджер зможуть зв’язатися з іншими членами команди щодо поправок у графік робіт або змін в грі через Jira та зможуть обговорити це все у текстовому або відеоформаті в Slack.

2.5. Ігрові рушії. Розповсюджені ігрові рушії

Під назвою ігрового рушія розуміють деякий додаток розробки програмного забезпечення, розроблений спеціально для створення відеоігор. Він містить

основну бібліотеку функцій, що використовуються в грі, однак вона існує незалежно від вмісту конкретної гри. Активи, які роблять гру унікальною, наприклад тема та персонажі, формують вміст. Ігровий рушій використовується для керування взаємодією ресурсів, щоб оживити гру. Це досягається за допомогою набору повторно використовуваних компонентів, які можуть служити двигуном для продуктів різних жанрів і стилів[20].

Роль ігрового рушія полягає в тому, щоб виконувати всю важку роботу за кадром, яка робить будь-яку відеогру придатною для гри. Центральні компоненти включають механізм візуалізації графіки, механізм фізики та виявлення зіткнень.

Механізм візуалізації генерує дво- або тривимірне зображення з файлу сцени, який визначає такі елементи, як точка огляду, освітлення та текстура. Фізичні механізми наближають рух реального світу у спосіб, звичний для гравця, але також значущий у контексті гри. Виявлення зіткнень передбачає те, як гра виявляє, відображає та реагує на перетин двох або більше візуалізованих об’єктів.

Інші ресурси, такі як мережеві можливості, штучний інтелект і створення звуку, також часто є компонентами ігрового движка. Як правило, компоненти доступні розробнику через візуальне інтегроване середовище розробки (IDE), яке спрощує створення гри. Наприклад, файли сцени можна прототипувати та тестувати за допомогою механізму візуалізації, не виходячи з IDE. Такий підхід також сприяє повторному використанню компонентів, що робить процес створення більш простим і ефективним.

Більшість перших ігрових рушіїв були власними проектами, розробленими для використання в одній грі. Гнучкість, що забезпечується вибором улюблених компонентів і створенням спеціалізованого інтерфейсу, була високо оцінена. Такий підхід давав розробникам великий контроль над зовнішнім виглядом ігор, але значно подовжував цикл розробки. Обмеження щодо вартості зазвичай забороняють використовувати власний ігровий движок у більшості проектів.

Доступні повні, готові до використання пакети для розробки ігор, які надають усі функціональні можливості запатентованих дизайнів. Багато з них є комерційними продуктами, але відкритий ігровий движок порівнянної якості

знайти неважко. Вони відкриті для налаштування коду, щоб відповідати звичним для розробника стилям або інструментам програмування. Рушій може не мати такої гнучкості, як пропрієтарний, але час і гроші, заощаджені на розробці, зазвичай значно переважують втрату контролю над ядром гри.

На наш час існує велика кількість рушіїв, як закритих, якими володіють студії-розробники, так і відкритих, як Unity, Godot і Unreal Engine, які будуть розглянуті далі.

Найпопулярнішим серед розробників ігор на основі відкритих рушіїв є Unity. Він пройшов довгий шлях від досить слабкого інструменту з вузьким функіоналом до досить сучасного та повного інструментів продукту. Unity — це кросплатформний ігровий рушій, розроблений Unity Technologies, який в основному використовується для розробки відеоігор і симуляцій для комп'ютерів, консолей і мобільних пристроїв. Вперше він був було оголошений ексклюзивом для OSX на Всесвітній конференції розробників Apple у 2005 році, проте з тих пір він розширилася до 27 платформ[21]. Він також був широко імплементований і використовується в інших галузях крім відеоігор, таких як кіноіндустрія, автомобільна промисловість, архітектура, інженерія та будівництво.

Прикладами ігор, написаних на Unity, є Battlestar Galactica Online, Firewatch, Unturned, Layers of Fear, Armello, The Long Dark і Traffic Racer. Крім того існують такі ігри, як Pillars of Eternity і Tyranny, які використовують модифіковану версію Unity, є рольовими іграми, створеними за допомогою цього рушія.

Головним конкурентом Unity є розробка Epic Games – Unreal Engine. На відміну від Unity, цей рушій був розроблений для розробки внутрішніх продуктів компанії. Unreal Engine – це суто тривимірне середовище розробки. Серед його переваг – потужні інструменти рендеру, виділення пам'яті та велика низка пов'язаних продуктів, які протягом життя проєкту купувала та інтегрувала в екосистему рушія команда Epic Games, такий як наприклад Quixel Mixer, продукт, який був по суті бібліотекою матеріалів та об'єктів, за допомогою яких можна було легко реалізувати художнє бачення будь-якої сцени[22].

На даний момент можна розробляти гру з двовимірними елементами на рушії за допомогою системи нашарування, коли кожен з шарів об’єктів ігрового простору накладається один на одного, як різні шари сцен в анімації чи в графічному програмному забезпеченні. Проте сам рушій не розрахований на це, і з імплементацією таких елементів може виникнути велика кількість проблем, які ще не були вирішені спільнотою розробників.

Рушій Unreal Engine спочатку був розроблений для використання в якості базової технології для відеоігор. Він використовується в ряді популярних ігор із високими графічними характеристиками, включаючи PlayerUnknown's Battlegrounds, Final Fantasy VII Remake, Valorant і Yoshi’s Crafted World, на додаток до ігор, розроблених ними, як Gears of War і Fortnite.

Останнім представником сучасних популярних відкритих рушіїв є Godot. Godot з самого початку розроблявся, як окремий продукт та платформа для необмеженого фінансовому в план виробництва ігор. Він надає менш широкий функціонал та набір функцій, проте він постійно розвивається і надає змогу зекономити час на розробці власних функцій. Як і Unity, він має пресети для як і двовимірних, так і тривимірних ігор.

Для вибору рушія важливу роль відіграє співвідношення ціни та якості. Усі три вище наведених варіантів є безкоштовними для використання, проте, якщо розробники збираються видавати і розповсюджувати свої розробки, частина прибутку піде до розробників рушія. Кожен із них надає свої прайсинги за надані функції. Unity пропонує три плани підписки, окрім безкоштовної, яка дозволить отримувати прибуток з продукту: Plus, Pro, і Enterprise. Ці плани різняться кількістю чистого прибутку, яку отримуватиме команда: Plus для прибутку нижче 200 тис. доларів і Pro – для прибутків вище. Unreal Engine пропонує лиш одну опцію ліцензування ціною в 1500 доларів на рік за особу, у якої, проте, немає обмежень на те, яким може бути максимальний прибуток з продажу продукту. Godot поширюється за системою, яка довзоляє повністю та без виплат розробникам рушія отримувати прибуток з відогри.

Для команди проєкту потрібно як мінімум п’ять ліцензій – для двох розробників, левел-дизайнера, графічного дизайнера та тестувальника. У результаті дослідження наданих функцій було вирішено використати рушій Unity через скалювання цін та опцій ліцензування та великої кількості документації та стороннього контенту.

2.6. Аналіз проєктів. Бенчмаркінг

Для того, щоб розробити успішний продукт, необхідно провести бенчмаркінг, тобто адаптацію найкращих практик, присутніх в успішних продуктах, схожих у своїй структурі до розроблюваного. Для виконання завдання будуть розглянуті три приклади ігор, зроблених незалежними студіями, які подібні до того, який буде вироблятися у процесі виконання проєкту: Hades, Hyper Light Drifter і Moon Hunters.

Hades (Supergiant, 2020). Найвідомішим з продуктів у даному списку є гра Hades. Розроблена студією-ветераном Supergiant на основі модифікованого Unreal Engine, ця гра показує, що робить ігри жанру rouge-like популярними та успішними. Серед ознак, які виділяють проєкт поміж інших, про які позитивно відзивалися користувачі є: унікальний сетинг, який виділяв його серед інших представників жанру, багате звукове оформлення з адаптивним саундтреком, бойова система, яка поступово вводила гравця в суть ігрових механік, висока реграбельність, унікальне поєднання дво- і трохвимірної графіки, м’яка система покарань за помилки, гнучка система обмежень та висока стеля для покращення своїх результатів.

Hyper Light Drifter (Heart Machine, 2016). Випущена в 2016 році незалежною студією Heart Machine, Hyper Light Drifter став першою грою команди, який зміг дати їй старт для подальших проєктів. Проєкт був профінансований через краудфандинг та розроблений на рушії Game Maker Studio. Серед відмінних ознак проєкту, про які позитивно відзивалися користувачі: ітеративна бойова система, побудована на комбінаціях, зміні

підходів та ухиляннях, середовищне оповідання, грандіозний саундтрек і дизайн середовищ.

Moon Hunters (Kitfox Games, 2016). Невеликий проєкт, розроблений з великим фокусом на використання процедурної генерації як елементу наративного дизайну. Ця гра є найменш популярною серед користувачів, проте вона виділяється деякими ознаками, які врешті зробили проєкт прибутковим, попри його неідеальність. Серед позитивних ознак проєкту користувачі відмічають: цікаву, хоч і погано збалансовану, ігрову систему, унікальний візуальний стиль гри, можливість кооперативного проходження рівнів, цікаві головоломки і генерацію рівнів.

У результаті проведеного дослідження можна зробити висновок про те, на яких елементах розробки треба зосередитися, та які так звані штампи жанру, яких очікує цільова аудиторія. Ці ознаки – це проста бойова система з великою кількістю розгалужень в дизайні, яка не перенавантажує нового гравця і не старому гравцеві різні методи досягнення мети гри, звукове і візуальне оформлення, за рахунок якого гру запам’ятають серед геймплейно схожих продуктів та генерація, яка може довго дивувати користувачів цікавими рівнями.

Виходячи в вище наведеної інформації, було прийняте рішення найняти експертів, здатних реалізувати елементи, які підвищать популярність продукту: маркетолога, аналітика ринку, звукового дизайнера і левел-дизайнера.

ВИСНОВОК ДО РОЗДІЛУ 2

1. Управління проєктами – це невід'ємна частина процесу розробки програмних продуктів.
2. Традиційні лінійні методології управління проєктами не підходять для більшості проєктів в галузі. Через занадто жорсткий розклад і важкість ведення паралельних процесів, була надана перевага іншим методам.
3. Серед популярних методів управління проєктів існують гнучкі методології Agile, які були розроблені з урахуванням тенденцій та ефективних методів управління в сфері інформаційних технологій.
4. Scrum – це найбільш доцільний метод для використання для цієї роботи через його специфіку – більшу визначеність ніж Kanban і контроль над роботами, використовуючи цінності Scrum. Він є одним із найбільш поширених способів для планування та проведення робіт, від спринта до спринта.
5. Для реалізації системи гнучкого управління буде необхідно застосувати декілька програмних рішень, для того щоб задовольняти принцими Scrum, пов'язаних з постійним спілкуванням і постійного внесення поправок відносно ситуації, яка буде складатися.
6. Для виконання роботи необхідно буде застосувати ігровий рушій для пришвидшення процесу розробки продукту. У результаті аналізу функціоналу, способів монетизації, доступної підтримки і документації був вибраний варіант розробки на Unity.
7. Були досліджені проєкти, схожі на виконуваний. У результаті були встановлені найбільш важливі елементи таких продуктів: цікаве оформлення, якісний музичний супровід, комплексний дизайн рівнів, механічно складна бойова система, яка при цьому не перевантажує гравця.
8. У результаті була сформована інформаційна база для подальшого успішного виконання роботи над проєктом.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ УПРАВЛІННЯМ ПРОЦЕСУ РОЗРОБКИ “RUNIC FOREST”. ОФОРМЛЕННЯ ПРОЄКТНОЇ ДОКУМЕНТАЦІЇ

3.1. Статут проєкту розробки комп’ютерної гри “Runic Forest”

Статут проекту (англ. project charter) — документ, розроблений ініціатором або спонсором проекту, який офіційно підтверджує існування проекту та надає керівнику проекту повноваження використовувати ресурси організації в операціях проекту[23]. Він є початковим документом етапу ініціації проекту разом із наказом про призначення команди проекту та аналізом зацікавлених сторін проекту. Далі буде розроблений статут проєкту.

Назва проєкту: Дизайн та контроль розробки комп’ютерної відеогри «Runic Forest».

Суть проєкту: Дизайн та створення розважального програмного забезпечення в жанрі rogue-like, який буде розроблений з урахуванням характеристик сучасного середньостатистичного комп’ютерного обладнання.

Цілі проєкту:

- Розробка процедурного генератора рівнів.
- Впровадження комплексного та захоплюючого ігрового циклу.
- Впровадження оригінального музичного супроводу.
- Імплементація мережевої підтримки для мультплеєрного режиму.

Продукт проєкту задовольнає потреби:

- Забезпечення користувача гнучким і довговічним джерелом розважального контенту
- Соціалізація у ігровій формі через мультиплеєр
- Адаптивна система рівнів

Бізнес вимоги:

- Збір статистики користувачів для аналізу
- Створення компетитивного вітчизняного ринку

- Продукт повинен надати матеріальну базу для подальшої діяльності студії

Функціональні вимоги:

- Можливість персоналізації стилю гри
- Візуальна кастомізація
- Вибір рівня складності
- Можливість сумісного проходження
- Можливість встановлення рекордів
- Можливість впливу гравця на варіативність проходження
- Приємний процес ігрового циклу

Нефункціональні вимоги:

- Доступ до ігор на платформах розповсюдження
- Збереження файлів гравця
- Мова програмування C#

Сегмент ринку: інтернаціональний ринок з розрахунком на аудиторію жанрів пригодницьких та екшн ігор.

3.1.1. Короткий опис проєкту

Проєкт розробки займається описом реалізації програмного забезпечення “Runic Forest”. Цей продукт являє собою комп’ютерну відеогру жанру roguelike, основною цінністю якого є унікальна частина геймплею, якою є механіка, яка дозволяє користувачам на відміну від інших представників жанру окрім факторів, які будуть збільшувати силу персонажу гравця під час ігрового процесу, дозволить вибирати труднощі, з якими він стикнеться під час проходження циклу за допомогою внутрішньоігрового елементу (так званих “рун”). Це надасть можливість для вищої реграбельності через збільшення кількості випадкових факторів за одне проходження та зменшить давлення на гравця, який матиме змогу підлаштовувати перешкоди на свій смак.

Основною стильовою частиною середовища внутрішньоігрового світу був вибраний сеттинг безкрайнього лісу. Через велику варіативність даної ідеї це відкриє для дизайнерів великий простір для кастомізації різних рівнів гри та ігрових елементів, як активних, так і пасивних.

За мету геймлейного циклу був взятий класичний для стилю ігровий процес, який полягає в тому, що гравець повинен пройти деяку кількість випадково згенерованих рівнів, отримувати випадкову винагороду за нього і у процесі дійти до останнього рівня та успішно завершити його. У процесі розробки буде прикута сильна увага до фактору перепроходження гри – для успішності продукту важливо забезпечити гравця цікавим способом провести велику кількість часу.

3.1.2. SWOT-аналіз проєкту

Для аналізу доцільності роботи над продуктом в практиці керівництва проєкту використовують так званий SWOT-аналіз. SWOT - це абревіатура від Strengths, Weaknesses, Opportunities, Threats. Іноді його також можна знайти під назвою як аналіз «WOTS up» або аналіз TOWS. Технологію приписують Альберту Хамфрі, який керував дослідницьким проектом у Стенфордському університеті в 1960-х і 1970-х роках, використовуючи дані провідних компаній, залучених до процесів довгострокового планування[24].

SWOT-аналіз — це інструмент планування, який використовується для розуміння ключових факторів — сильних і слабких сторін, можливостей і загроз — залучених до проекту чи організації. Це включає формулювання мети організації чи проекту та ідентифікацію внутрішніх і зовнішніх факторів, які або підтримують, або несприятливі для досягнення цієї мети. SWOT часто використовується як частина стратегічного процесу або процесу планування, але його можна застосувати, щоб допомогти зрозуміти організацію чи ситуацію, а також для прийняття рішень для багатьох різних сценаріїв.

Цінність SWOT полягає головним чином у тому, що він пропонує самооцінку для керівництва. Методологія має перевагу в тому, що вона

використовується одночасно як швидкий інструмент або комплексний інструмент управління, і що один може призвести до іншого. Ця гнучкість є одним із факторів, які сприяли його успіху.

Однак, незважаючи на те, що елементи можуть здаватися оманливо простими та легкими у застосуванні, досвід показує, що ефективний і значущий SWOT-аналіз потребує часу та значних ресурсів. Вирішити, якими є сильні та слабкі сторони організації, а також оцінити вплив і ймовірність можливостей і загроз набагато складніше, ніж здається на перший погляд. Це вимагає командних зусиль і не може бути ефективно виконане лише однією людиною.

Для розробки форми для аналізу такого типу для проєкту в даній роботі були використані дані, приведені в теоретичному пункті. Проєкт був проаналізований з точки зору SWOT-аналізу, беручи до уваги специфіку локального ринку та стану ігрової індустрії вцілому на даний час. Наприкінці був отриманий результат дослідження, приведений у таблиці 3.1.

Таблиця 3.1

Сильні сторони	Слабкі сторони
• Сучасний продукт з можливістю виходу на мультиплатформний рівень • Зручний інтерфейс та система управління • Гнучкі системи варіативності ігрового процесу • Оригінальне візуальне та звукове оформлення • Доступні інструменти розробки та документація	• Невеликий інтерес до інвестування в галузь на локальному рівні • Слабко розвинений ринок спеціалістів • Ексклюзивність для однієї платформи • Залежність від сервісів ігрових майданчиків • Проблеми з видавництвом

Закінчення табл. 3.1

Можливості	Загрози
• Розвиток ігрового ринку в Україні • Залучення експертів, ексклюзивних для галузі на локальному ринку • Подальший розвиток команди проєкту та розробка нових продуктів з прибутку • Підвищення кількості іноземних інвестицій	• Відсутність джерел фінансування та брак інших матеріальних ресурсів • Недостатній інтерес до вітчизняного продукту • Піратство контенту та відсутність захисту • Продукт може не окупитися

Після проведення аналізу були визначені слабкі сторони та загрози для продукту. Підсумовуючи, зроблено висновок, що у продукту є велика кількість викликів, проте сильні сторони проєкту переважають їх. Через великий потенціал IT-індустрії в Україні, можна сказати, що залучення молодих експертів до слабко розвинених, проте досить прибуткових галузей в сфері інформаційних технологій, дозволить більшій кількості людей працювати в цьому напрямі. Створення цього продукту, особливо правильне його планування, створить більше кількість прецедентів та надасть досвід майбутнім на сучасним експертам для подальшої розробки ігрових продуктів на локальному ринку, враховуючи його специфіку. У результаті, чим більша буде кількість цікавих проєктів, тим менше будуть ставати ризики через підвищення інтересу до такого типу продуктів.

Для того, щоб забезпечити майбутнє здоров’я галузі необхідно робити експерименти, аналізувати їх результати та постійно імплементувати нові та загальноприйняті на світовому рівні практики, які стосуються жанрів та стилів оформлення, з якими працює кожна студія. Сильна сторона будь-якого продукту –

це бажання покупця знов і знов повертатися до нього, чого можна досягнути за допомогою грамотного підходу до планування процесів та розуміння стану ринку.

Підбиваючи підсумки, SWOT-аналіз показав, що сильні сторони є запорукою його успіху, у той час, як слабкі сторони можна нівелювати розвитком галузі, частиною процесу якого і розроблюваний продукт.

3.1.3. WBS-структура проєкту

Для того, щоб почати проводити роботу над проєктом потрібно задати питання про те, які саме роботи потрібно виконувати для виконання проєкту.

У даній роботі буде використаний фреймворк Scrum, який розрахований на планування роботи в так звані спринти: невеликі відрізки часу, під час яких виконуються певні задачі, задані загальним планом робіт. Часто ці роботи пов’язані в спільний комплекс задач, які над паралельно працюють різні відділи компанії.

У таблиці 3.1 приведені контрольні точки спринтів в проєкті.

Таблиця 3.2

Код	Контрольна віха	Часові рамки
1	Стадія планування	01.02.23 – 14.03.23
2	Розробка гри - спринт 1	15.03.23 – 30.03.23
3	Розробка гри - спринт 2	01.04.23 – 20.04.23
4	Розробка гри - спринт 3	21.04.23 – 10.05.23
5	Розробка гри - спринт 4	11.05.23 – 01.06.23
6	Розробка гри - спринт 5	02.06.23 – 22.06.23
7	Розробка гри - спринт 6	23.06.23 – 13.07.23
8	Стадія контролю якості	24.07.23 – 19.09.23
9	Реліз продукту	20.09.23 – 18.10.23

Work Breakdown Structure або WBS – це ієрархічна модель робіт, яку широко використовують в управлінні проєктами. Основною задачею цієї схеми є

показати розбір великих робіт на менші підпункти за допомогою морфологічної декомпозиції великих процесів[25]. Нижче наведена система WBS проєкту:

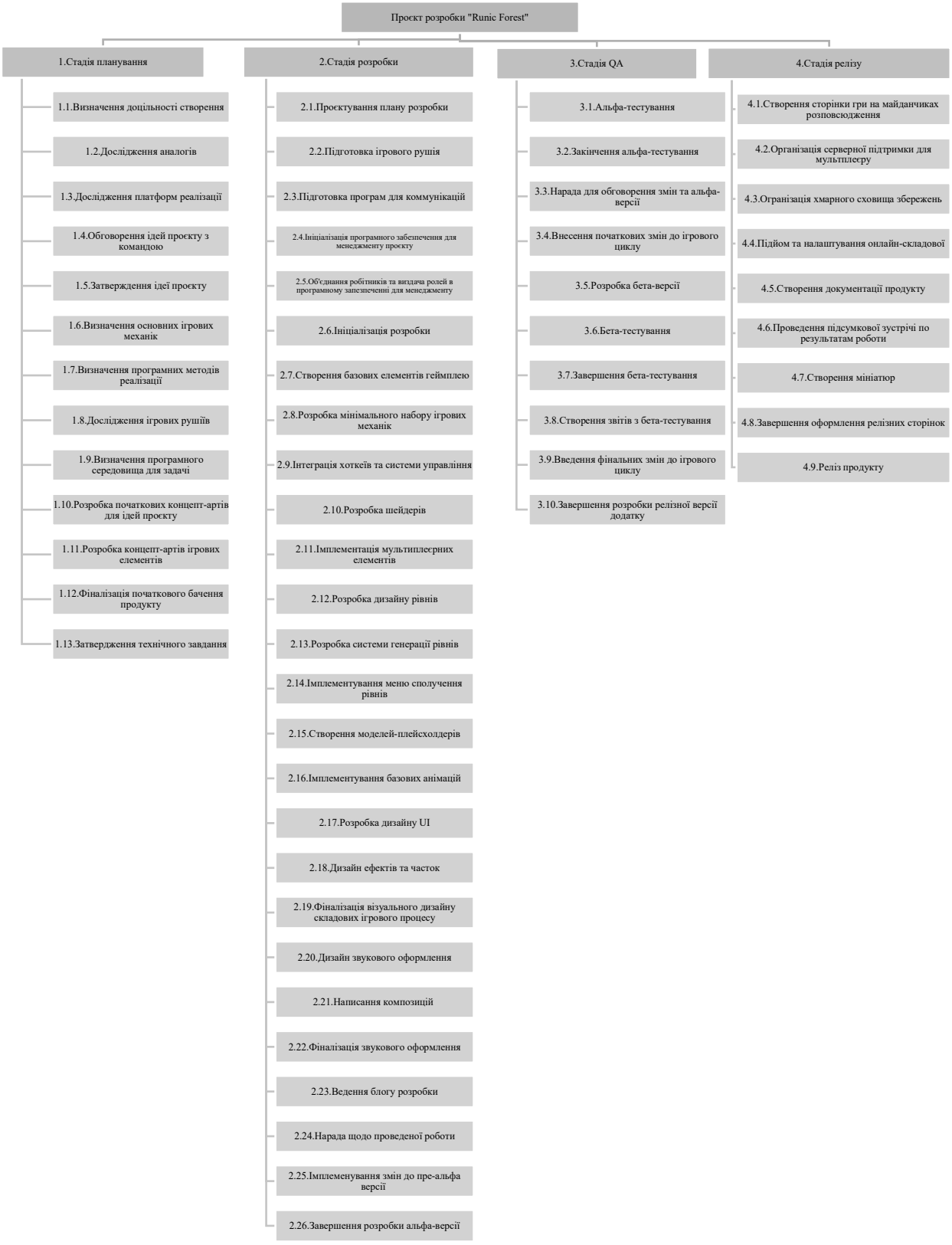


Рис. 3.1. Ієрархічна модель організаційної структури проєкту

3.1.4. OBS-структура команди проєкту

Для виконання будь-якого проєкту створюється певна команда осіб, яка повинна відповідати за реалізацію задач, поставлених у плані.

З точки зору проєктного управління, команда – це тимчасове підприємство, яке об'єднує у собі осіб або їх груп, залучених до виконання задач роботи над проєктом і несуть відповідальність перед керівцтвом за їх виконання. Тимчасовість організації полягає в тому, що певна команда збирається окремо для кожного проєкту окремо.

Для формування команд використовують широку низку підходів, які показують відповідальність і обов'язки, які потрубно задовольнити у процесі роботи над продуктом. Проте не зважаючи на вибраний метод залишається принцип субординації: члени команди і її керівники підпорядковуються і звітують перед керівником проєкту. Він має оцінювати здатність спеціалістів до виконання поставлених задач та організації структури їх роботи. У випадку великої кількості підпорядкованого персоналу, він може визначити ключових відповідальний осіб, які будуть відповідати за свої відділи чи підрозділи.

Для роботи в невеликій команді треба працювати з абсолютним мінімумом професійних ресурсів, які можуть бути доступні для неї. Причиною для такого кроку є те, що створення ігор – це досить комплексний процес, який потребує багато часу. Часто ігри є продуктом захоплення членів команди деякою ідеєєю, яка повинна буди реалізована у результаті проведеної роботи.

Запорукою успішного просування компанії та розвитку внутрішьої культури є чітке розуміння обов'язків кожним членом команди та прийняття будь-яких думок, пропрацьовування ідей кожного члена коллективу, якщо в них з'являються ідеї. Не всі ідеї прийматимуть для фінального продукту, та це створює творчу атмосферу в студії. Проте найголовнішим фактором є атмосфера довіри, яка і повинна створитися у результаті дотримання вище приведених засад.

OBS-структура проєкту – це ієрархічна модель, яка структурована так, щоб допомогти визначити, які працівники відповідатимуть за певні частини проекту.

Це інструмент, який часто використовується для керування встановленою робочою структурою, встановленою в рамках проекту[26].

На рисунку 3.2 показана організація команди проєкту у вигляді OBS.

Рис. 3.2. Ієрархічна модель організаційної структури проєкту

Команда проєкту складатиметься з нижче наведених спеціалістів:

- Проєктний менеджер
- Програміст 1
- Програміст 2
- Саунд-дизайнер

- Графічний дизайнер
- Левел-дизайнер
- Тестувальник
- Системний адміністратор
- Системний аналітик
- Ігровий аналітик
- Менеджер спільноти
- Маркетолог
- Керівник проєкту

3.1.5. Стейкхолдери проєкту

Проект не може бути реалізований однією людиною. Незалежно від розміру проекту, до виконання завдань, пов'язаних із проектом, залучено кілька осіб та організацій. Такі особи та організації визначаються як зацікавлені сторони. Щоб досягти цілей і завдань даного проекту, менеджери проекту повинні розуміти свої відносини з зацікавленими сторонами. Тому визначення зацікавлених сторін проекту є важливим аспектом управління проектом.

Зацікавлену сторону можна визначити як будь-яку особу чи організацію, які безпосередньо беруть участь у певному проекті. Крім того, стейкхолдер також можна визначити як особу, інтереси якої можуть позитивно чи негативно вплинути на проект. Як наслідок, існують різні типи зацікавлених сторін, які можуть бути залучені до конкретного проекту. Крім того, ці зацікавлені сторони постійно відрізняються для різних видів проектів.

Керівники проекту співпрацюють із зацікавленими сторонами проекту, щоб вести проект від його початку до завершення. Оскільки існує кілька типів зацікавлених сторін проекту з різними рівнями зацікавленості в проекті, до визначення зацікавлених сторін проекту слід підходити з самого початку проекту[27]. Цей процес визначення зацікавлених сторін в управлінні проектом

має вирішальне значення для розуміння участі кожної зацікавленої сторони проекту.

Зазвичай процес визначення зацікавлених сторін є першим кроком, і зацікавлені сторони проекту згадуються в статуті проекту або будь-якому іншому документі планування проекту. Цей документ міститиме повний список зацікавлених сторін проекту на початку проекту. Це також включатиме деяких зацікавлених сторін, які беруть участь у проекті на пізніших етапах життєвого циклу проекту. Як керівник проекту, ви несете відповідальність передбачити ці зацікавлені сторони проекту та окреслити їхню участь у проекті до їх фактичної участі.

У даному проєкті будуть залучені таки види стейкхолдерів:

- Зовнішні:
    1. Спонсори проєкту, а саме тут – учасники краудфандингової компанії. Вони мають влив на те, наскільки сильно може змінюватися проєкт під час розробки та часто їх залучають до обгооврення введення нових ідей через спеціальні канали зв’язку.
    2. Користувачі, тобто покупці фінального продукту. Вони не мають вплив на хід проєкту через відсутність доступу до фінального продукту. Вплив йде лише після публікації продукту і може вплинути на майбутні проєкти команди.
- Внутрішні
    1. Проєктний менеджер. Проєктний менеджер суттєво впливає на проєкт і є одним із ключових стейкхолдерів. Він визначає план дій, координує роботу відділів, проводить аналіз проєктів і забезпечує зв’язок між стейкхолдерами.
    2. Команда проєкту. Вона досить сильно впливає на хід проєкту, так як напряму зацікавлена в реалізації проєктних робіт монетарною інсентивою. Експерти можуть допомогти з процесом ведення розробки пропонуючи нові ідеї чи оптимізуючи свою роботу через автоматизацію процесів.

3.1.6. Обмеження проєкту

Обмеження проекту – це будь-які фактори, які певним чином стримують проект. Вони можуть бути внутрішніми або зовнішніми і зазвичай належать до однієї з чотирьох категорій: час, вартість, обсяг або якість.

Важливо зазначити, що не всі обмеження є негативними. Фактично, деякі з них можуть бути навіть корисними для проекту. Наприклад, обмеженням може бути те, що вам дозволено використовувати лише певні матеріали або що команда повинна дотримуватися певного стилю розробки.

У будь-якому випадку, ключовим моментом є виявлення обмежень на ранній стадії та розробка плану, як із ними впоратися якомога ефективніше.

Серед обмежень проєкту є наступні:

Таблиця 3.3

Код	Назва задачі	Тривалість задачі	Рамки дат
1.	Старт проєкту	0 днів	01.02.23
2.	Стадія планування	30 днів	01.02.23-14.03.23
3.	Стадія розробки	60 днів	15.03.23-06.06.23
4.	Стадія QA	75 днів	07.06.23-19.09.23
5.	Стадія релізу	21 день	20.09.23-17.10.23
6.	Завершення проєкту	0 днів	18.10.23

Обмеженнями щодо організації робіт є:

- Мова програмування C#
- Використання ліцензійного програмного забезпечення
- Використання рушія Unity

Бюджетне обмеження складає 1 100 000 гривень. Ця сума складається з загальної вартості проєкту, встановленої під час проєктування - 859 130 гривень, а також з резервного фонду у 240 000 гривень.

3.2. Управління проєктом розробки “Runic Forest”

3.2.1. План управління комунікаціями

Для того, щоб підтримувати зв’язок між різними стейкхолдерами проєкту, проєктному менеджеру потрібно визначити план їх взаємодії через звіти та зустрічі. Часто як гарну практику застосовують протоколювання. Нижче наведені методи такої взаємодії.

Таблиця 3.4

Тип комунікацій	Учасники	Привід комунікацій	План типу комунікацій	Час комунікації	Відповідальний
Відкриваюча зустріч	Робоча команда проєкту	Обговорення організаційних питань, мети проєкту і його ідей	Очна зустріч	На початку проєкту	Керівник проєкту
Звітність за станом виконання	Публікується для учасників краудфандингової компанії	Сповіщення донатерів про стан продукту	Сайт розробника, блог	Кожен четвер	Керівник проєкту
Переписування про проєкт	Зацікавлені сторони проєкту	Робочі питання	Slack	При необхідності	Ініціатор
Робочі зустрічі	Робоча команда проєкту	Робочі питання і методи усунення проблем	Slack	При необхідності	Керівник проєкту
Підсумкова зустріч	Робоча команда проєкту	Підбиття підсумків проведеної роботи та обговорення даних з бета-тестів	Очна зустріч	Наприкінці проєкту	Керівник проєкту

3.2.2. План управління термінами

Перед тим, як починати визначати завтрати на проєкт, необхідно провести аналіз часу, який піде на виконання робіт. Крім того, планування цих робіт і зв’язків між ними допоможе приймати рішення щодо змін в проєкті, якщо буде проблема з браком часу для виконання робіт.

Найрозповсюдженіший метод аналізу термінів роботи є таблиця задач. Вона визначає затрати в часі, які потрібні для кожного етапу чи роботи. Крім того, ефективність методу вирішення задач з часом показана в наглядності такого підходу, де кожній роботі відповідає часовий ресурс, доступний для виконання[28].

Для внесення змін у початковий план або залучення не зайнятих спеціалістів існує визначення задачі-попередника. Так можна розбити проєкт на різні ланцюжки задач, за які відповідають різні підрозділи команди. У результаті можна регулювати роль та продуктивність кожного з них.

У таблиці 3.5 приведений план робіт з урахуванням їх зв’язків та часу, який потрібен для їх виконання.

Таблиця 3.5

Код	Назва задачі	Тривалість задачі	Задача попередник
1	**2**	**3**	**4**
1.	**Стадія планування**	**30 днів**	**-**
1.1.	Визначення доцільності створення	1 день	-
1.2.	Дослідження аналогів	3 дні	Визначення доцільності створення
1.3	Дослідження платформ реалізації	3 дні	Дослідження аналогів
1.4.	Обговорення ідей проєкту з командою	2 дні	Дослідження платформ реалізації
1.5.	Затвердження ідеї проєкту	5 днів	Обговорення ідей проєкту з командою
1.6.	Визначення основних ігрових механік	5 днів	Обговорення ідей проєкту з командою
1.7.	Визначення програмних методів реалізації	5 днів	Визначення основних ігрових механік

Продовження табл. 3.5

1	2	3	4
1.8.	Дослідження ігрових рушіїв	3 дні	Затвердження ідеї проєкту, Визначення основних ігрових механік, Визначення програмних методів реалізації
1.9.	Визначення програмного середовища для задачі	1 день	Дослідження ігрових рушіїв
1.10.	Розробка початкових концепт-артів для ідей проєкту	5 днів	Обговорення ідей проєкту з командою
1.11.	Розробка концепт-артів ігрових елементів	5 днів	Розробка початкових концепт-артів для ідей проєкту
1.12.	Фіналізація початкового бачення продукту	3 дні	Розробка концепт-артів ігрових елементів, Дослідження ігрових рушіїв
1.13.	Затвердження технічного завдання	5 днів	Фіналізація початкового бачення продукту, Визначення програмного середовища для задачі
2.	**Стадія розробки**	**60 днів**	**-**
2.1.	Проєктування плану розробки	5 днів	Затвердження технічного завдання
2.2.	Підготовка ігрового рушія	5 днів	Проєктування плану розробки
2.3.	Підготовка програм для коммунікацій	2 дні	Проєктування плану розробки
2.4.	Ініціалізація програмного забезпечення для менеджменту проєкту	3 дні	Підготовка програм для коммунікацій
2.5.	Об'єднання робітників та виздача ролей в програмному запезпеченні для менеджменту	2 дні	Ініціалізація програмного забезпечення для менеджменту проєкту
2.6.	Ініціалізація розробки	1 день	Ініціалізація програмного забезпечення для менеджменту проєкту, Підготовка ігрового рушія
2.7.	Створення базових елементів геймплею	5 днів	Ініціалізація розробки
2.8.	Розробка мінімального набору ігрових механік	5 днів	Створення базових елементів геймплею
2.9	Інтеграція хоткеїв та системи управління	2 дні	Розробка мінімального набору ігрових механік
2.10.	Розробка шейдерів	3 днів	Інтеграція хоткеїв та системи

			управління

Продовження табл. 3.5

1	2	3	4
2.11.	Імплементація мультиплеєрних елементів	5 днів	Розробка шейдерів
2.12.	Розробка дизайну рівнів	10 днів	Імплементація мультиплеєрних елементів
2.13.	Розробка системи генерації рівнів	10 днів	Імплементація мультиплеєрних елементів
2.14.	Імплементування меню сполучення рівнів	5 днів	Розробка дизайну рівнів, Розробка системи генерації рівнів
2.15.	Створення моделей-плейсхолдерів	2 днів	Розробка концепт-артів ігрових елементів, Дослідження ігрових рушіїв
2.16.	Імплементування базових анімацій	3 дні	Фіналізація початкового бачення продукту, Визначення програмного середовища для задачі
2.17.	Розробка дизайну UI	5 днів	-
2.18.	Дизайн ефектів та часток	5 днів	Затвердження технічного завдання
2.19.	Фіналізація візуального дизайну складових ігрового процесу	15 днів	Проєктування плану розробки
2.20.	Дизайн звукового оформлення	5 днів	Проєктування плану розробки
2.21.	Написання композицій	20 днів	Підготовка програм для коммунікацій
2.22.	Фіналізація звукового оформлення	5 днів	Ініціалізація програмного забезпечення для менеджменту проєкту
2.23.	Ведення блогу розробки	45 днів	Ініціалізація програмного забезпечення для менеджменту проєкту, Підготовка ігрового рушія
2.24.	Нарада щодо проведеної роботи з тестової версії	1 день	Ініціалізація розробки
2.25.	Імплеменування змін до пре-альфа версії	10 днів	Створення базових елементів геймплею
2.26.	Завершення розробки альфа-версії	1 день	Розробка мінімального набору ігрових механік
3.	**Стадія QA**	**75**	**-**
3.1.	Альфа-тестування	7	Завершення розробки альфа-версії
3.2.	Закінчення альфа-тестування	1	Альфа-тестування
3.3.	Нарада для обговорення змін	1	Закінчення альфа-тестування

	та альфа-версії		

Закінчення табл. 3.5

1	2	3	4
3.4.	Внесення початкових змін до ігрового циклу	11 днів	Нарада для обговорення змін та альфа-версії
3.5.	Розробка бета-версії	15 днів	Внесення початкових змін до ігрового циклу
3.6.	Бета-тестування	15 днів	Розробка бета-версії
3.7.	Завершення бета-тестування	15 днів	Бета-тестування
3.8.	Створення звітів з бета-тестування	1 день	Завершення бета-тестування
3.9.	Введення фінальних змін до ігрового циклу	5 днів	Створення звітів з бета-тестування
3.10.	Завершення розробки релізної версії додатку	5 днів	Введення фінальних змін до ігрового циклу, Створення звітів з бета-тестування
4.	**Стадія релізу**	**21 днів**	**-**
4.1.	Створення сторінки гри на майданчиках розповсюдження	3 дні	Завершення розробки релізної версії додатку
4.2.	Організація серверної підтримки для мультплеєру	3 дні	Створення сторінки гри на майданчиках розповсюдження
4.3.	Огранізація хмарного сховища збережень	5 днів	Організація серверної підтримки для мультплеєру
4.4.	Підйом та налаштування онлайн-складової	3 день	Огранізація хмарного сховища збережень
4.5.	Створення документації продукту	1 день	Підйом та налаштування онлайн-складової
4.6.	Проведення підсумкової зустрічі по результатам роботи	3 день	Створення документації продукту
4.7.	Створення мініатюр	1 день	Створення сторінки гри на майданчиках розповсюдження
4.8.	Завершення оформлення релізних сторінок	1 день	Створення мініатюр
4.9.	Реліз продукту	-	Завершення оформлення релізних сторінок, Ведення блогу розробки, Проведення підсумкової зустрічі по результатам роботи

3.2.3. План управління ресурсами

Після створення плану робіт в часі, потрібно розпланувати проєкт з точки зору персоналу, який працює над проєктом. Управління ресурсами – це основний процес, який відбувається увесь час виконання робіт через людський фактор та непередбачувані обставини. Часто часові обмеження пливають на баланс людських ресурсів, задіяних у виконанні поставленої задачі.

Балансуючи між витратами в часі та доступністю того чи іншого професійного ресурсу, проєктний менеджер має змогу коригувати план роботи на ходу, замість того, щоб переписувати план проєкту з нуля, враховуючи обставини, які можуть скластися.

Однією з функцій створення подібної таблиці є знаходження боттлнеків – вузьких місць – тих частин процесів, з якими працівник або не може впоратися, або він потребує більше ресурсів для нівелювання негативного впливу зовнішніх чинників[29].

За допомогою таблиці 3.6 можна визначити кількість співробітників, задіяних в кожному з цих завдань та їх ролі.

Таблиця 3.6

Код	Назва задачі	Тривалість задачі	Ресурс
1	**2**	**3**	**4**
1.	**Стадія планування**	**30 днів**	-
1.1.	Визначення доцільності створення	1 день	Ігровий аналітик; Керівник проєкту; Проєктний менеджер; Системний аналітик; Маркетолог
1.2.	Дослідження аналогів	3 дні	Ігровий аналітик; Системний аналітик; Маркетолог
1.3	Дослідження платформ реалізації	3 дні	Ігровий аналітик; Програміст 1; Програміст 2; Системний аналітик
1.4.	Обговорення ідей проєкту з командою	2 дні	Графічний дизайнер; Левел-дизайнер; Менеджер спільноти;Програміст 1; Програміст 2; Проєктний менеджер; Саунд-дизайнер;Тестувальник

Продовження табл. 3.6

1	2	3	4
1.5.	Затвердження ідеї проєкту	5 днів	Керівник проєкту; Проєктний менеджер
1.6.	Визначення основних ігрових механік	5 днів	Левел-дизайнер; Програміст 1; Програміст 2
1.7.	Визначення програмних методів реалізації	5 днів	Ігровий аналітик; Програміст 1; Програміст 2; Проєктний менеджер
1.8.	Дослідження ігрових рушіїв	3 дні	Програміст 1; Програміст 2; Саунд-дизайнер
1.9.	Визначення програмного середовища для задачі	1 день	Програміст 1; Програміст 2
1.10.	Розробка початкових концепт-артів для ідей проєкту	5 днів	Графічний дизайнер
1.11.	Розробка концепт-артів ігрових елементів	5 днів	Графічний дизайнер
1.12.	Фіналізація початкового бачення продукту	3 дні	Ігровий аналітик; Проєктний менеджер; Системний аналітик
1.13.	Затвердження технічного завдання	5 днів	Проєктний менеджер; Системний адміністратор
2.	**Стадія розробки**	**60 днів**	-
2.1.	Проєктування плану розробки	5 днів	Проєктний менеджер
2.2.	Підготовка ігрового рушія	5 днів	Програміст 1; Програміст 2
2.3.	Підготовка програм для коммунікацій	2 дні	Системний адміністратор
2.4.	Ініціалізація програмного забезпечення для менеджменту проєкту	3 дні	Проєктний менеджер; Системний адміністратор
2.5.	Об'єднання робітників та виздача ролей в програмному запезпеченні для менеджменту	2 дні	Графічний дизайнер; Левел-дизайнер; Програміст 1; Програміст 2; Проєктний менеджер; Саунд-дизайнер
2.6.	Ініціалізація розробки	1 день	Програміст 1; Програміст 2
2.7.	Створення базових елементів геймплею	5 днів	Програміст 1; Програміст 2;Тестувальник
2.8.	Розробка мінімального набору ігрових механік	5 днів	Програміст 1; Програміст 2; Левел-дизайнер

Продовження табл. 3.6

1	2	3	4
2.9	Інтеграція хоткеїв та системи управління	2 дні	Програміст 1; Програміст 2; Левел-дизайнер
2.10.	Розробка шейдерів	3 днів	Програміст 1;Програміст 2
2.11.	Імплементація мультиплеєрних елементів	5 днів	Левел-дизайнер;Програміст 1;Програміст 2;Тестувальник
2.12.	Розробка дизайну рівнів	10 днів	Левел-дизайнер;Програміст 1
2.13.	Розробка системи генерації рівнів	10 днів	Програміст 2;Тестувальник
2.14.	Імплементування меню сполучення рівнів	5 днів	Програміст 1;Програміст 2
2.15.	Створення моделей-плейсхолдерів	2 днів	Графічний дизайнер
2.16.	Імплементування базових анімацій	3 дні	Графічний дизайнер
2.17.	Розробка дизайну UI	5 днів	Графічний дизайнер
2.18.	Дизайн ефектів та часток	5 днів	Графічний дизайнер
2.19.	Фіналізація візуального дизайну складових ігрового процесу	15 днів	Графічний дизайнер
2.20.	Дизайн звукового оформлення	5 днів	Саунд-дизайнер
2.21.	Написання композицій	20 днів	Саунд-дизайнер
2.22.	Фіналізація звукового оформлення	5 днів	Саунд-дизайнер
2.23.	Ведення блогу розробки	45 днів	Менеджер спільноти
2.24.	Нарада щодо проведеної роботи з тестової версії	1 день	Графічний дизайнер;Левел-дизайнер;Програміст 1;Програміст 2;Проєктний менеджер;Саунд-дизайнер
2.25.	Імплеменування змін до пре-альфа версії	10 днів	Графічний дизайнер;Левел-дизайнер;Програміст 1;Програміст 2;Саунд-дизайнер
2.26.	Завершення розробки альфа-версії	1 день	Графічний дизайнер;Керівник проєкту;Левел-дизайнер;Програміст 1;Програміст 2;Саунд-дизайнер
3.	**Стадія QA**	**75**	**-**
3.1.	Альфа-тестування	7	Програміст 1; Програміст 2; Тестувальник

Закінчення табл. 3.6

1	2	3	4
3.2.	Закінчення альфа-тестування	1	Програміст 1; Програміст 2; Проєктний менеджер; Тестувальник
3.3.	Нарада для обговорення змін та альфа-версії	1	Графічний дизайнер; Керівник проєкту; Левел-дизайнер; Програміст 1; Програміст 2; Саунд-дизайнер
3.4.	Внесення початкових змін до ігрового циклу	11 днів	Левел-дизайнер; Програміст 1; Програміст 2
3.5.	Розробка бета-версії	15 днів	Графічний дизайнер; Левел-дизайнер; Програміст 1; Програміст 2
3.6.	Бета-тестування	15 днів	Тестувальник
3.7.	Завершення бета-тестування	15 днів	Програміст 1; Програміст 2; Проєктний менеджер; Тестувальник
3.8.	Створення звітів з бета-тестування	1 день	Проєктний менеджер
3.9.	Введення фінальних змін до ігрового циклу	5 днів	Графічний дизайнер; Левел-дизайнер; Програміст 1; Програміст 2
3.10.	Завершення розробки релізної версії додатку	5 днів	Графічний дизайнер; Керівник проєкту; Левел-дизайнер; Програміст 1; Програміст 2; Саунд-дизайнер; Тестувальник
4.	**Стадія релізу**	**21 днів**	**-**
4.1.	Створення сторінки гри на майданчиках розповсюдження	3 дні	Менеджер спільноти; Маркетолог
4.2.	Організація серверної підтримки для мультплеєру	3 дні	Програміст 1; Програміст 2; Системний адміністратор
4.3.	Огранізація хмарного сховища збережень	5 днів	Програміст 1; Програміст 2; Системний аналітик
4.4.	Підйом та налаштування онлайн-складової	3 день	Програміст 1; Програміст 2
4.5.	Створення документації продукту	1 день	Проєктний менеджер
4.6.	Проведення підсумкової зустрічі по результатам роботи	3 день	Графічний дизайнер; Керівник проєкту; Програміст 1; Програміст 2; Проєктний менеджер; Саунд-дизайнер; Тестувальник
4.7.	Створення мініатюр	1 день	Графічний дизайнер
4.8.	Завершення оформлення релізних сторінок	1 день	Маркетолог; Менеджер спільноти
4.9.	Реліз продукту	-	Керівник проєкту;Проєктний менеджер

3.2.4. План управління вартістю

Матеріальні обмеження проєкту – ще один з видів ресурсів, управлінням якого займається проєктний менеджер. Часто відхилення відхилення у витратах є прогнозованим явищем та відбувається у практиці увесь час. Головне, що повинен зробити менеджер у випадку неспівставлення запланованих витрат з фактичнимим – це провести правильний комплекс заходів, який пом'якшить втрати та підвищить відхилення в позитивну сторону.

Контроль вартості проєкту складається з моніторингу вартісниз показників імплеиентації проєкту, динамічної зміни бюджету, своєчасного інформування стейкхолдерів про поточний матеріальний стан проєкту та запобіганню приняттю хибних рішень[30].

Для цього проєкту був розроблений план робіт з урахуванням їх матеріального ресурсу. У результаті в таблиці 3.7 були представлені ресурси з їх відповідними витратами.

Таблиця 3.7

Код	Ресурс	Тип ресурсу	Вартість ресурсу
1	2	3	4
1.	Проєктний менеджер	Трудовий	170,00 грн/год
3.	Програміст 1	Трудовий	150,00 грн/год
4.	Програміст 2	Трудовий	150,00 грн/год
5.	Саунд-дизайнер	Трудовий	120,00 грн/год
6.	Графічний дизайнер	Трудовий	150,00 грн/год
7.	Левел-дизайнер	Трудовий	100,00 грн/год
8.	Тестувальник	Трудовий	100,00 грн/год
9.	Системний адміністратор	Трудовий	100,00 грн/год
10.	Системний аналітик	Трудовий	120,00 грн/год
11.	Ігровий аналітик	Трудовий	100,00 грн/год
12.	Менеджер спільноти	Трудовий	90,00 грн/год

Закінчення табл. 3.7

1	2	3	4

13.	Маркетолог	Трудовий	120,00 грн/год
14.	Ліцензійне ПЗ	Матеріальний	80 000,00 ₴ грн
15.	Інтернет-з'єднання	Матеріальний	1 350,00 ₴ грн/рік
16.	Хостинг	Матеріальний	2 000,00 ₴ грн/рік

Підсумувавши питомі витрати на кожен ресурс, були визначені витрати на кожен етап окремо так підбита очікувана вартість реалізації проєкту. Результати приведені у таблиці 3.8.

Таблиця 3.8

Код	Назва задачі	Тривалість задачі	Вартість етапу, грн
1.	Стадія планування	30 днів	131 440
2.	Стадія розробки	60 днів	398 730
3.	Стадія QA	75 днів	268 080
4.	Стадія релізу	21 день	60 880
5.	Усього	-	859 130

У результаті була отримана сума в 859 130 гривень. Для того, щоб забезпечити стабільність роботи системи, потрібно виділити ще певну суму грошей, за рахунок якої можна буде пом’якшити додаткові витрати. У результаті складений бюджет дорівнюватиме 1 200 000 гривень.

3.2.5. План управління ризиками

Ризик – це будь-яка несподівана подія, яка може вплинути на проект – на краще чи на гірше. Ризик може вплинути на будь що: на людей, процеси, технології та ресурси. Важливо пам’ятати, що ризики – це не те саме, що проблеми. Проблеми – це речі, з якими ви знаєте, що вам доведеться мати справу, і можете навіть уявити, коли вони виникнуть, як-от запланована відпустка члена команди або значне зростання попиту на товари під час свят. Ризики – це події,

які можуть відбутися, і ніхто може не знати, коли, наприклад, сезон грипу, який відразу вразить команду, або ключовий компонент продукту незамовлений[31].

Для проєкту був розроблений список ризиків, які можуть спіткати команду проєкту і представлений у таблиці 3.9.

Таблиця 3.9

№	Найменування ризику	Ймовірність виникнення	Вплив ризику
1	Зрив темпу робіт	0.3	0.8
2	Недостатній рівень підготовки персоналу	0.4	0.4
3	Збільшення витрат під час виконання	0.7	0.4
4	Звільнення ключових працівників	0.1	0.1
5	Недостатнє фінансування	0.2	0.8
6	Проблеми комунікації в команді	0.4	0.4
7	Помилки в формулюванні завдань проєкту	0.2	0.8
8	Вибір неоптимальної схеми реалізації розробки продукту	0.1	0.8

Для аналізу нагальності реагування на ці ризики була створена таблиця критичності ризиків (табл. 3.10):

Таблиця 3.10

Ймовірність					
0.8 – 1.0					
0.6 – 0.8				**3**	
0.4 – 0.6					
0.2 – 0.4				**2, 6**	**1**
0.0 – 0.2		**4**			**5, 7, 8**
	0.05	**0.1**	**0.2**	**0.4**	**0.8**
	Вплив				

Для боротьби з ризиками в галузі управління проєктами існує два способи: запобігання, тобто попередження ризику, і усунення, якщо превентивні заходи не

змогли зупинити його появу. Методи, які будуть застосовані в цьому проєкті приведені в таблиці 3.11.

Таблиця 3.11

№	Ризик	Попередження ризику	Усунення ризику
1	**2**	**3**	**4**
1.	Зрив темпу робіт	Моніторинг виконання проєкту, перепланування затримок	Виділення потужностей на проблемні задачі, залучення зовнвшньох підтримки
2.	Недостатній рівень підготовки персоналу	Якісний відбір кандидатів на посади, професійна перевірка їх знань та навичок	Пошук нового персоналу
3.	Збільшення витрат під час виконання	Врахувати додаткові витрати під час планування бюджету, розробити детальний план витрат для проєкту	Мінімізація витрат через перерозподілення грошових ресурсів
4.	Звільнення ключових працівників	Якісний відбір кандидатів на посади, набір	Заміна члена команди, по можливості переведення частини інших працівників на завдання
5.	Недостатнє фінансування	Просування продукту, ведення блогу розробки, залучення платників через спеціальні пропозиції	Залучення кредитних коштів, мінімізація втрат через пререрозподілення коштів
6.	Проблеми комунікації в команді	Чіткий регламент робіт, запровадження професйної етики на робочому місці	Проведення наради по проблемним питанням, усунення конфліктів через технічні фактори, реогранізація команди
7.	Помилки в формулюванні завдань проєкту	Перевірка статуту проєкту і проєктної документації, складання детального плану проєкту, визначення вимог та відповідальностей працівникків	Збільшення часових рамок проєкту та виправлення неточностей
8.	Вибір неоптимальної схеми реалізації розробки продукту	Детальний аналіз можливостей, слабких і сильних сторін кожного з доступних варіантів, використання досвіду індустрії, відкритість обговорення деталей проєкту	Рефоматування методів розробки, перехід на інші фреймворки розробки

ВИСНОВОК ДО РОЗДІЛУ 3

1. Продуктом проєкту розробки є відеогра “Runic Forest”.

2. Цілі проєкту розробки “Runic Forest”: розробка процедурного генератора рівнів, впроваждення комплексного та захоплюючого ігрового циклу, впровадження оригінального музичного супроводу, імплементація мережевої підтримки для мультплеєрного режиму.

3. Обмеження проєкту: дата завершення роботи над проєктом: 18.10.23, виставлено обмеження загального бюджету: 1 100 000 гривень. Серед обмежень організації робіт: використання рушія Unity, використання ліцензійного програмного забезпечення, мова програмування C#

4. Команда проєкту складається з проєкту розробки “Runic Forest”: Проєктний менеджер, Програміст 1, Програміст 2, Саунд-дизайнер, Графічний дизайнер, Левел-дизайнер, Тестувальник, Системний адміністратор, Системний аналітик, Ігровий аналітик, Менеджер спільноти, Маркетолог.

5. Стейкхолдери проєкту включають в себе: фінансувальники і спонсори проєкту – краудфандингові патрони, користувачі: гравці “Runic Forest”, організаційні групи: менеджер проєкту і команнда проєкту, зовнішня консультаційна група,

6. На виконання проєкту виділено 186 днів та 859 130 гривень. Для резерву було прийняте рішення виділити додаткові кошти, 240 000 гривень.

7. У результаті був комплексно проаналізований процес роботи над продуктом і отриманий чіткий план розробки програмного забезпечення “Runic Forest”.

4.

ВИСНОВКИ

1. Ігрова індустрія – це досить прибутковий і переспективний бізнес, на сцені якого Україна представлена досить слабко. Освоєння цього ринку зможе привнести в країну велику кількість іноземних інвестицій та сучасних практик розробки програмного забезпечення.

2. Галузь розробки комп'ютерних ігор містить у собі велику кількість професій, ексклюзивних для виробництва розважального програмного забезпечення, тож розширення виробництва ігор розширить можливості працевлаштування людей в галузі інформаційних технологій

3. Галузь проєктного управління є невід'ємною частиною сучасного світу інформаційних технологій. За допомогою практик управління проєктами можна налагодити роботу комплексних програмних рішень

4. В галузі розробки програмних продуктів не існує універсальних методів розробки продуктів. Кожна підгалузь та тип продукту може використовувати різні підходи до планування робіт.

5. Найпоширенішими фреймворками для розробки програмного забезпечення є ті, які були побудовані на засадах гнучкої розробки Agile. Найпопулярнішими з них є Scrum і Kanban.

6. Були досліджені проєкти, схожі на виконуваний. У результаті були встановлені найбільш важливі елементи таких продуктів: цікаве оформлення, якісний музичний супровід, комплексний дизайн рівнів, механічно складна бойова система, яка при цьому не перевантажує гравця.

7. Цілі проєкту розробки "Runic Forest": розробка процедурного генератора рівнів, впроваждення комплексного та захоплюючого ігрового циклу, впровадження оригінального музичного супроводу, імплементація мережевої підтримки для мультплеєрного режиму.

8. Продуктом проєкту розробки є відеогра "Runic Forest".

9. Обмеження проєкту: дата завершення роботи над проєктом: 18.10.23, виставлено обмеження загального бюджету: 1 100 000 гривень. Серед обмежень організації робіт: використання рушія Unity, використання ліцензійного програмного забезпечення, мова програмування C#.

10. На виконання проєкту виділено 186 днів та 859 130 гривень. Для резерву було прийняте рішення виділити додаткові кошти, 240 000 гривень.

11. У результаті був комплексно проаналізований процес роботи над продуктом і отриманий чіткий план розробки програмного забезпечення “Runic Forest”.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Hines V. The state of project management 2020 [Електронний ресурс] / Vince Hines. – [Б. м. : б. в.], 2021. – Режим доступу: https://wellingtone.co.uk/wp-content/uploads/2020/06/The-State-of-Project-Management-Report-2020-Wellingtone.pdf (дата звернення: 27.11.2022).
2. What is a videogame? Rules, puzzles and simulations: defining the object of study [Електронний ресурс] // Videogames. – [Б. м.], 2004. – С. 19–38. – Режим доступу: https://doi.org/10.4324/9780203642900-5 (дата звернення: 10.12.2022).
3. Пономаренко І. В. Рекламна монетизація в ігровій індустрії [Електронний ресурс] : Thesis / Пономаренко Ігор Віталійович, Ярема К. О. – [Б. м.], 2021. – Режим доступу: https://er.knutd.edu.ua/handle/123456789/18651 (дата звернення: 10.12.2022).
4. Bateman C. "Meet Bertie the Brain, the world's first arcade game, built in Toronto" / Chris Bateman // Spacing magazine.
5. Wolf M. J. P. The video game explosion: A history from PONG to playstation and beyond / Mark J. P. Wolf. – [S. l.] : Greenwood Press, 2007. – 400 p.
6. Совгиря Р. П. Методи та інструменти удосконалення управління проєктами в організації в сучасних умовах [Електронний ресурс] : Магістерська робота / Совгиря Роман Павлович. – [Б. м.], 2021. – Режим доступу: https://dspace.znu.edu.ua/jspui/handle/12345/6577 (дата звернення: 10.12.2022)
7. Ergasheva S. Software development life cycle early phases and quality metrics: a systematic literature review [Електронний ресурс] / Shokhista Ergasheva, Artem Kruglov // Journal of physics: conference series. – 2020. – Т. 1694. – С. 012007.
8. равченко Г. Ю. Адаптивне управління: можливості та специфіка адаптивного управління розвитком [Електронний ресурс] / Ганна Юріївна Кравченко // V міжнародна науково-практична конференція «АКТУАЛЬНІ

ПИТАННЯ ОСВІТИ І НАУКИ». – [Б. м.], 2017. – Режим доступу: https://doi.org/10.26697/9786177089000.2017.78 (дата звернення: 10.12.2022).

9. Беляченко В. В. Управління ризиками створення елементів автоматизованих систем управління [Електронний ресурс] / В. В. Беляченко, С. В. Бобров, М. К. Утюшев // Збірник наукових праць Центру воєнно-стратегічних досліджень НУОУ імені Івана Черняховського. – 2021. – № 3-70. – С. 101–106.
10. Hazzan O. The Agile Manifesto [Електронний ресурс] / Orit Hazzan, Yael Dubinsky // Agile Anywhere. – Cham, 2014. – С. 9–14.
11. Бушуєв С. Д. Методологія, методи і засоби проектного менеджменту. Практика проектного менеджменту «крок за кроком», методичні вказівки з питань занять: для студентів спеціальності «Проектний менеджмент». С.Д. Бушуєв.– КНУБА, 1999. – 34 с.
12. Burachek I. Scrum as a successful innovative method of project management [Електронний ресурс] / Igor Burachek, Olha Zakapko, Dina Yarmolyk // Market infrastructure. – 2021. – № 51.
13. Чорний А. В. Роль Скрам-майстра в розвитку лідерських компетентностей персоналу IT-підприємств / А. В. Чорний // Бізнес Інформ. – 2019. – № 1. – С. 383–395.
14. Локк, Д. Основи управління проектами/Д. Локк; пров. з англ. М.: – HIPPO, 2014. – 253 с.
15. Кондратюк Д. М. Оптимальний постачальник як чинник конкурентоспроможності підприємств / Д. М. Кондратюк. // Економіка. Управління. Інновації. – 2018. – 5 с.
16. Домашенко С. В. Інформаційні технології в управлінні підприємством: електронний документообіг / С. В. Домашенко // Збірник наукових праць Таврійського державного агротехнологічного університету. Економічні науки. – 2013. – № 2 (22), т. 3. – С. 103–112.
17. Ha A. Atlassian launches A marketplace for project management add-ons [Електронний ресурс] / Anthony Ha // TechCrunch. – Режим доступу:

https://techcrunch.com/2012/05/30/atlassian-marketplace/ (дата звернення: 10.12.2022). – Назва з екрана.
18.3Smith C. Slack statistics and facts (2020) | by the numbers [Електронний ресурс] / Craig Smith. – Режим доступу: https://web.archive.org/web/20200617181227/https://expandedramblings.com/index.php/slack-statistics/.
19. Bouquin D. R. GitHub [Електронний ресурс] / Daina R. Bouquin // Journal of the medical library association : jMLA. – 2015. – Т. 103, № 3. – С. 166–167. – Режим доступу: https://doi.org/10.3163/1536-5050.103.3.019 (дата звернення: 10.12.2022).
20. Game Engines [Електронний ресурс] / Jonas Freiknecht [та ін.] // Serious Games. – Cham, 2016. – С. 127–159. – Режим доступу: https://doi.org/10.1007/978-3-319-40612-1_6 (дата звернення: 10.12.2022).
21. Brodkin J. How Unity3D Became a Game-Development Beast [Електронний ресурс] / Jon Brodkin // Dice Insights. – Режим доступу: https://www.dice.com/career-advice/how-unity3d-become-a-game-development-beast (дата звернення: 10.12.2022).
22. Ізотова Н. Художній наратив у контексті ігрової стилістики [Електронний ресурс] / Наталя Ізотова // Humanities science current issues. – 2019. – Т. 1, № 20. – С. 63–67.
23. Pmbok [Електронний ресурс] // Encyclopedia of education and information technologies. – Cham, 2020. – С. 1258.
24. Грей К.Ф. Управління проектами: Практичний посібник: Пер. з англ. / К.Ф. Грей, Е.У. Ларсон – М.: «Справа і Сервіс», 2019. – 57 с.
25. Бушуєв С. Д. Креативні технології управління проектами та програмами / С. Д. Бушуєв, Н. С. Бушуєва, І. А. Бабаєв [та ін.] – К. : «Саміт-Книга«, 2010. – 2-3 с.
26. Попов Ю.І. Управління проектами: Навч. допомога. / Ю.І.Попов, О.В. Яковенко – М.: «ІНФРА-М», 2018. – 34 с

27. овгиря Р. П. Методи та інструменти удосконалення управління проєктами в організації в сучасних умовах [Електронний ресурс] : Магістерська робота / Совгиря Роман Павлович. – [Б. м.], 2021. – Режим доступу: https://dspace.znu.edu.ua/jspui/handle/12345/6577
28. Зуб А. Т. Управління проектами: підручник і практикум для академічного бакалавріату / А. Т. Зуб. – М.: «Видавництво Юрайт», 2014. – 422-438 с.
29. Краснобаєва А. Д. Особливості управліннння персоналом в сучасних організаціях [Електронний ресурс] : Theses / Краснобаєва А. Д. – [Б. м.], 2014.
30. Ципі. Г. Менеджмент проектів у практиці сучасної компанії / Г. Ципес, А. Товб. – М.: Олімп-Бізнес, 2016. – 304 с.
31. Корж Б. В. Гнучкі моделі управління командною роботою інжинірингових проєктів [Електронний ресурс] : Thesis / Корж Б. В., Приймак В. М. – [Б. м.], 2020.

ДОДАТКОК

Додаток 1

Відстежування процесів в проєкті в MS Project

Стадія планування		**30 дней**	**Ср 01.02.23**	**Вт 14.03.23**
Визначення доцільності створення	Ігровий аналітик;Кері	1 день	Ср 01.02.23	Ср 01.02.23
Дослідження аналогів	Ігровий аналітик;Сист	3 дней	Чт 02.02.23	Пн 06.02.23
Дослідження платформ реалізації	Ігровий аналітик;Прог	3 дней	Вт 07.02.23	Чт 09.02.23
Обговорення ідей проекту з командою	Графічний дизайнер;/	2 дней	Пт 10.02.23	Пн 13.02.23
Затвердження ідеї проекту	Керівник проекту;Прс	5 дней	Вт 14.02.23	Пн 20.02.23
Визначення основних ігрових механік	Левел-дизайнер;Прог	5 дней	Вт 14.02.23	Пн 20.02.23
Визначення програмних методів реалізації	Ігровий аналітик;Прог	5 дней	Вт 21.02.23	Пн 27.02.23
Дослідження ігрових рушіїв	Програміст 1;Програм	3 дней	Вт 28.02.23	Чт 02.03.23
Визначення програмного середовища для задачі	Програміст 1;Програм	1 день	Пт 03.03.23	Пт 03.03.23
Розробка початкових концепт-артів для ідей проекту	Графічний дизайнер	5 дней	Вт 14.02.23	Пн 20.02.23
Розробка концепт-артів ігрових елементів	Графічний дизайнер	5 дней	Вт 21.02.23	Пн 27.02.23
Фіналізація початкового бачення продукту	Ігровий аналітик;Прог	3 дней	Пт 03.03.23	Вт 07.03.23
Затвердження технічного завдання	Проєктний менеджер;	5 дней	Ср 08.03.23	Вт 14.03.23

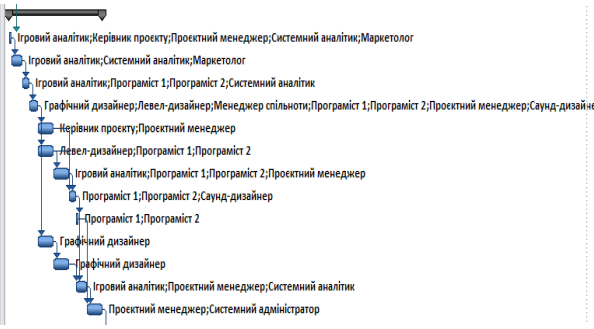


Рис. Д1.1

Стадія розробки		**60 дней**	**Ср 15.03.23**	**Вт 06.06.23**
Проєктування плану розробки	Проєктний менеджер;	5 дней	Ср 15.03.23	Вт 21.03.23
Підготовка ігрового рушія	Програміст 1;Програм	5 дней	Ср 22.03.23	Вт 28.03.23
Підготовка програм для коммунікацій	Системний адміністра	2 дней	Ср 22.03.23	Чт 23.03.23
Ініціалізація програмного забезпечення для менеджменту проекту	Проєктний менеджер;Системний	3 дней	Пт 24.03.23	Вт 28.03.23
Ініціалізація розробки	Графічний дизайнер;/	1 день	Ср 29.03.23	Ср 29.03.23
Створення базових елементів геймплею	Програміст 1;Програм	5 дней	Чт 30.03.23	Ср 05.04.23
Розробка мінімального набору ігрових механік	Програміст 1;Програм	5 дней	Чт 06.04.23	Ср 12.04.23
Інтеграція хоткеїв та системи управління	Програміст 1;Програм	2 дней	Чт 13.04.23	Пт 14.04.23
Розробка шейдерів	Програміст 1;Програм	5 дней	Пн 17.04.23	Пт 21.04.23
Імплементація мультиплеєрних елементів	Левел-дизайнер;Прог	5 дней	Пн 24.04.23	Пт 28.04.23
Розробка дизайну рівнів	Левел-дизайнер;Прог	10 дней	Пн 01.05.23	Пт 12.05.23
Розробка системи генерації рівнів	Програміст 2;Тестува	10 дней	Пн 01.05.23	Пт 12.05.23
Імплементування меню сполучення рівнів	Програміст 1;Програм	5 дней	Пн 15.05.23	Пт 19.05.23
Створення моделей-плейсхолдерів	Графічний дизайнер	2 дней	Чт 30.03.23	Пт 31.03.23
Імплементування базових анімацій	Графічний дизайнер	3 дней	Пн 03.04.23	Ср 05.04.23
Розробка дизайну UI	Графічний дизайнер	5 дней	Чт 06.04.23	Ср 12.04.23
Дизайн ефектів та часток	Графічний дизайнер	5 дней	Чт 13.04.23	Ср 19.04.23
Фіналізація візуального дизайну складових ігрового процесу	Графічний дизайнер	15 дней	Чт 20.04.23	Ср 10.05.23
Дизайн звукового оформлення	Саунд-дизайнер	5 дней	Чт 30.03.23	Ср 05.04.23
Написання композицій	Саунд-дизайнер	20 дней	Пн 10.04.23	Пт 05.05.23
Фіналізація звукового оформлення	Саунд-дизайнер	5 дней	Пн 08.05.23	Пт 12.05.23
Ведення блогу розробки	Менеджер спільноти	45 дней	Чт 30.03.23	Ср 31.05.23
Нарада щодо проведеної роботи з тестової версії	Графічний дизайнер;/	1 день	Пн 22.05.23	Пн 22.05.23
Імплементування змін до пре-альфа версії	Графічний дизайнер;/	10 дней	Вт 23.05.23	Пн 05.06.23
Завершення розробки альфа-версії	Графічний дизайнер;/	1 день	Вт 06.06.23	Вт 06.06.23

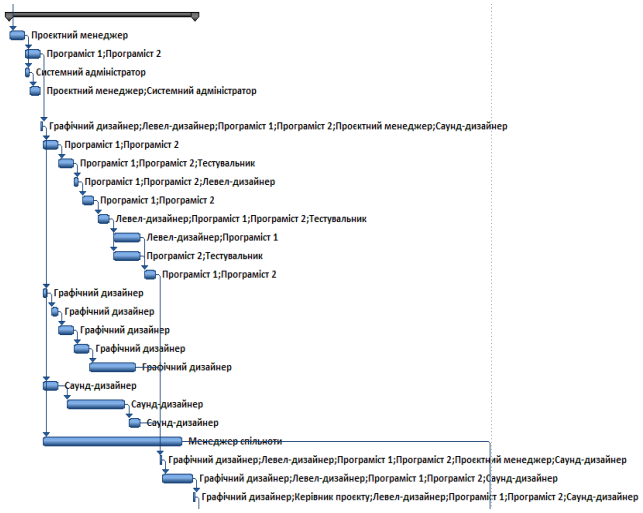


Рис. Д1.2

Стадія QA		**75 дней**	Ср (
Альфа-тестування	Програміст 1;Програм	7 дней	Ср (
Закінчення альфа-тестування	Програміст 1;Програм	1 день	Пт 1
Нарада для обговорення змін та альфа-версії	Графічний дизайнер;/	1 день	Пн
Внесення початкових змін до ігрового циклу	Левел-дизайнер;Прог	11 дней	Вт 2
Розробка бета-версії	Графічний дизайнер;/	15 дней	Ср (
Бета-тестування	Тестувальник	15 дней	Ср 2
Завершення бета-тестування	Програміст 1;Програм	15 дней	Ср 1
Створення звітів з бета-тестування	Проєктний менеджер	1 день	Ср (
Введення фінальних змін до ігрового циклу	Графічний дизайнер;/	5 дней	Ср (
Завершення розробки релізної версії додатку	Графічний дизайнер;/	5 дней	Ср 1
Стадія релізу		**21 дней**	Ср 1

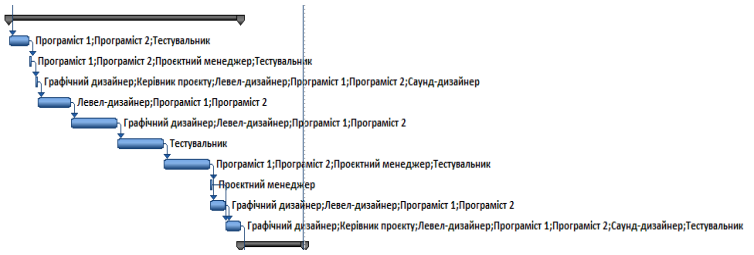


Рис. Д1.3

Стадія релізу		**21 дней**	Ср 20.09.23	Ср 18.10.23	
Створення сторінки гри на майданчиках розповсюдження	Менеджер спільноти;	3 дней	Ср 20.09.23	Пт 22.09.23	53
Організація серверної підтримки для мультплеєру	Програміст 1;Програм	3 дней	Пн 25.09.23	Ср 27.09.23	55
Організація хмарного сховища збережень	Програміст 1;Програм	5 дней	Чт 28.09.23	Ср 04.10.23	56
Підйом та налаштування онлайн-складової	Програміст 1;Програм	5 дней	Чт 05.10.23	Ср 11.10.23	57
Створення документації продукту	Проєктний менеджер;	3 дней	Чт 12.10.23	Пн 16.10.23	58
Проведення підсумкової зустрічі по результатам роботи	Графічний дизайнер;І	1 день	Вт 17.10.23	Вт 17.10.23	59
Створення мініатюр	Графічний дизайнер	3 дней	Пн 25.09.23	Ср 27.09.23	55
Завершення оформлення релізних сторінок	Маркетолог;Менедж	1 день	Чт 28.09.23	Чт 28.09.23	61
Реліз продукту	Керівник проєкту;Прс	1 день	Ср 18.10.23	Ср 18.10.23	39;62;60
Кінець проєкту		0 дней	Ср 18.10.23	Ср 18.10.23	63

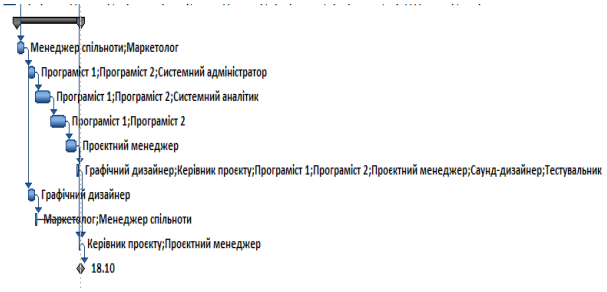


Рис. Д1.4

Додаток 2

Список ресурсів в проєкті в MS Project

	Название ресурса	Краткое название	Тип	Единицы измерения материалов	Макс. единиц	Стандартная ставка	Начисление	Базовый календарь	Макс. единиц	Ставка сверхурочных
1	Проєктний менеджер	П	Трудовой		100%	150,00 ₴/ч	Пропорциональное	Стандартный	100%	0,00 ₴/ч
2	Програміст 1	П	Трудовой		100%	150,00 ₴/ч	Пропорциональное	Стандартный	100%	0,00 ₴/ч
3	Програміст 2	П	Трудовой		100%	150,00 ₴/ч	Пропорциональное	Стандартный	100%	0,00 ₴/ч
4	Саунд-дизайнер	С	Трудовой		100%	120,00 ₴/ч	Пропорциональное	Стандартный	100%	0,00 ₴/ч
5	Графічний дизайнер	Г	Трудовой		100%	150,00 ₴/ч	Пропорциональное	Стандартный	100%	0,00 ₴/ч
6	Левел-дизайнер	Л	Трудовой		100%	100,00 ₴/ч	Пропорциональное	Стандартный	100%	0,00 ₴/ч
7	Тестувальник	Т	Трудовой		100%	100,00 ₴/ч	Пропорциональное	Стандартный	100%	0,00 ₴/ч
8	Системний адміністратор	С	Трудовой		100%	100,00 ₴/ч	Пропорциональное	Стандартный	100%	0,00 ₴/ч
9	Системний аналітик	С	Трудовой		100%	120,00 ₴/ч	Пропорциональное	Стандартный	100%	0,00 ₴/ч
10	Ігровий аналітик	І	Трудовой		100%	100,00 ₴/ч	Пропорциональное	Стандартный	100%	0,00 ₴/ч
11	Менеджер спільноти	М	Трудовой		100%	90,00 ₴/ч	Пропорциональное	Стандартный	100%	0,00 ₴/ч
12	Маркетолог	М	Трудовой		100%	120,00 ₴/ч	Пропорциональное	Стандартный	100%	0,00 ₴/ч
13	Ліцензійне ПЗ	Л	Материальный	од.		60 000,00 ₴	Пропорциональное			
14	Інтернет-з'єднання	І	Материальный	од.		1 350,00 ₴	Пропорциональное			
15	Хостинг	Х	Материальный	од.		2 000,00 ₴	Пропорциональное			
16	Домен	Д	Материальный	од.		500,00 ₴	Пропорциональное			
17	Бухгалтер	Б	Трудовой		100%	0,00 ₴/ч	Пропорциональное	Стандартный	100%	0,00 ₴/ч
18	Керівник проєкту	К	Трудовой		100%	0,00 ₴/ч	Пропорциональное	Стандартный	100%	0,00 ₴/ч

Рис. Д2.1

Додаток 3

Презентація

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
БУДІВНИЦТВА І АРХІТЕКТУРИ
Факультет автоматизації і інформаційних технологій
Кафедра управління проєктами

Атестаційна робота на здобуття освітнього ступеня магістра на тему:

"Управління проєктом розробки комп'ютерної гри"

Виконав: студент 6 курсу, групи КНУП-61
Степанов Л.П.
Керівник: доктор технічних наук, професор
Козир Б.Ю.

м. Київ — 2022

Рис. Д3.1

Опис предмету дослідження

Предмет дослідження: ігрова індустрія та методики розробки комп'ютерних ігор.

Теоретична та методологічна основа дослідження: для роботи над предметом дослідження були використані вітчизняні та іноземні статті, статистичні звіти галузевих компаній. Також використані витримки з різних професійних джерел, таких як PMBOK та Agile Alliance, та доповіді працівників лідерів галузі на тематичних професійних конференціях.

Практичне значення: основною метою проведеної роботи повинні стати загальні вказівки та практична база створення розважального програмного забезпечення. Також, робота покликана вивчити сучасні методи їх розробки, створення оформлення та просування, що допоможе галузі набрати обертів на вітчизняному ринку та вивести країну на більш конкурентний рівень на світовому ринку.

2

Рис. Д3.2

Задачі проєкту

1. Провести аналіз принципів та сталих в процесі розробки розважального програмного забезпечення;

2. Провести дослідження тенденцій внутрішнього ігрового ринку України та дослідити успішні світові практики;

3. Визначити специфічні ознаки роботи в галузі та імплементувати;

4. На основі отриманих даних визначити цінності та методологію для реалізації проєкту;

5. Створити план та проєктну документацію, за якою буде проведена робота над проєктом.

3

Рис. Д3.3

Опис проєкту

Цілі проєкту:

•Розробка процедурного генератора рівнів.

•Впровадження комплексного та захоплюючого ігрового циклу.

•Впровадження оригінального музичного супроводу.

•Імплементація мережевої підтримки для мультплеєрного режиму.

Продукт проєкту задовольнає потреби:

•Забезпечення користувача гнучким і довговічним джерелом розважального контенту

•Соціалізація у ігровій формі через мультиплеєр

•Адаптивна система рівнів

Бізнес вимоги:

• Збір статистики користувачів для аналізу

• Створення компетитивного вітчизняного ринку

• Продукт повинен надати матеріальну базу для подальшої діяльності студії

4

Рис. Д3.4

Ієрархічна структура робіт (WBS)

Рис. Д3.5

Обмеження проєкту

Назва задачі	Тривалість задачі	Вартість етапу, грн	Назва задачі
Стадія планування	30 днів	131 440	Стадія планування
Стадія розробки	60 днів	398 730	Стадія розробки
Стадія QA	75 днів	268 080	Стадія QA
Стадія релізу	21 день	60 880	Стадія релізу
Усього	-	859 130	Усього

Фінансовим обмеженням є бюджет проєкту, який становить 860 000 гривень.

Для створення резерву було прийняте рішення збільшити фінальний бюджет на 240 000 гривень.

Остаточний бюджет: 1 100 000 гривень.

Обмеженнями щодо організації робіт є:

- Використання ліцензійного програмного забезпечення
- Використання рушія Unity
- Використання мови програмування C#, як основного в Unity

Рис. Д3.6

SWOT – аналіз

Сильні сторони

- Сучасний продукт з можливістю виходу на мультиплатформний рівень
- Зручний інтерфейс та система управління
- Гнучкі системи варіативності ігрового процесу
- Оригінальне візуальне та звукове оформлення
- Доступні інструменти розробки та документація

Можливості

- Розвиток ігрового ринку в Україні
- Залучення експертів, ексклюзивних для галузі на локальному ринку
- Подальший розвиток команди проєкту та розробка нових продуктів з прибутку
- Підвищення кількості іноземних інвестицій

Слабкі сторони

- Невеликий інтерес до інвестування в галузь на локальному рівні
- Слабко розвинений ринок спеціалістів
- Ексклюзивність для однієї платформи
- Залежність від сервісів ігрових майданчиків
- Проблеми з видавництвом

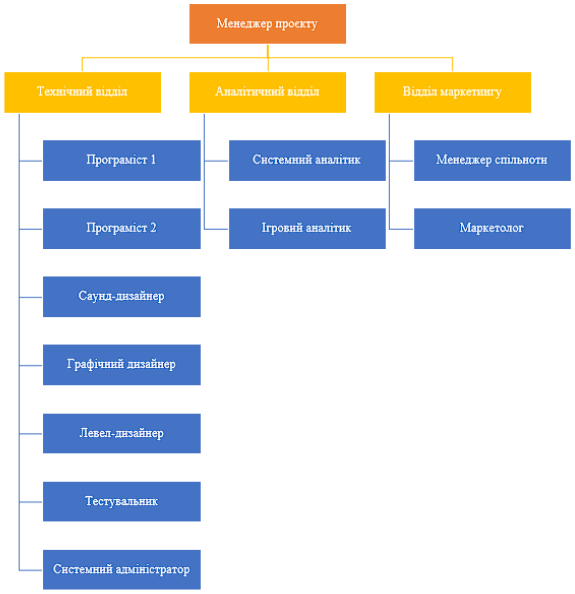
Загрози

- Відсутність джерел фінансування та брак інших матеріальних ресурсів
- Недостатній інтерес до вітчизняного продукту
- Піратство контенту та відсутність захисту
- Продукт може не окупитися

7

Рис. Д3.7

Ієрархічна структура організації (OBS)



8

Рис. Д3.8

Зацікавлені сторони проєкту

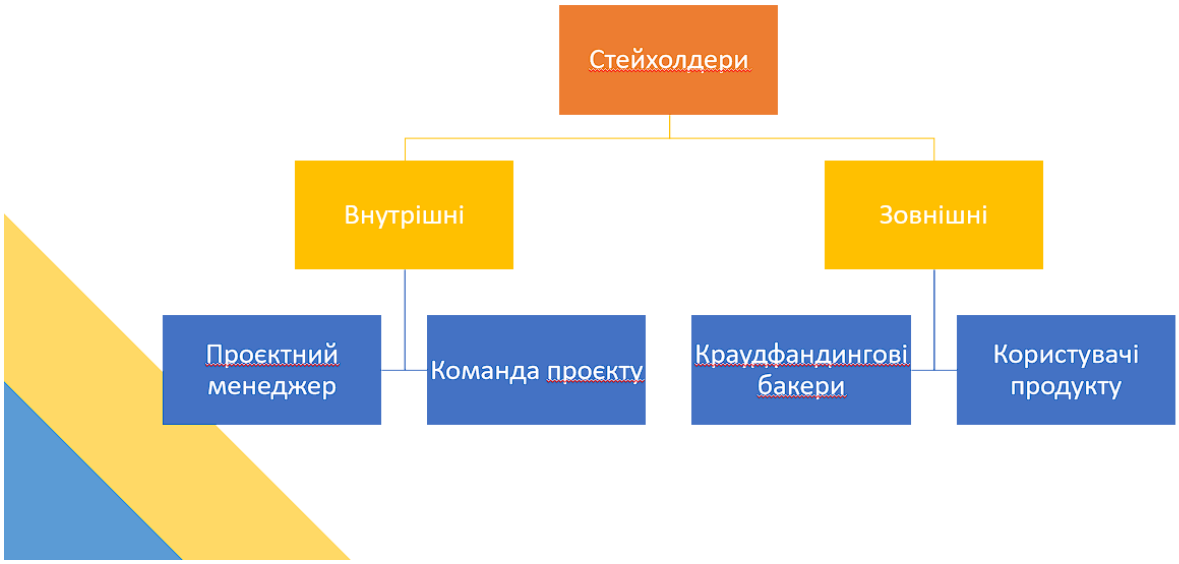


Рис. Д3.9

План комунікацій

Тип комунікацій	Учасники	Привід комунікацій	План типу комунікацій	Час комунікації	Відповідальний
Відкриваюча зустріч	Робоча команда проєкту	Обговорення організаційних питань, мети проєкту і його ідей	Очна зустріч	На початку проєкту	Керівник проєкту
Звітність за станом виконання	Публікується для учасників краудфандингової компанії	Сповіщення донатерів про стан продукту	Сайт розробника, блог	Кожен четвер	Керівник проєкту
Переписування про проєкт	Зацікавлені сторони проєкту	Робочі питання	Slack	При необхідності	Ініціатор
Робочі зустрічі	Робоча команда проєкту	Робочі питання і методи усунення проблем	Slack	При необхідності	Керівник проєкту
Підсумкова зустріч	Робоча команда проєкту	Підбиття підсумків проведеної роботи та обговорення даних з бета-тестів	Очна зустріч	Наприкінці проєкту	Керівник проєкту

Рис. Д3.10

План управління термінами

Код в WBS	Назва задачі	Час виконання	Попередник
2.11.	Імплементація мультиплеєрних елементів	5 днів	Розробка шейдерів
2.12.	Розробка дизайну рівнів	10 днів	Імплементація мультиплеєрних елементів
2.13.	Розробка системи генерації рівнів	10 днів	Імплементація мультиплеєрних елементів
2.14.	Імплементування меню сполучення рівнів	5 днів	Розробка дизайну рівнів, Розробка системи генерації рівнів
2.15.	Створення моделей-плейсхолдерів	2 днів	Розробка концепт-артів ігрових елементів, Дослідження ігрових рушіїв
2.16.	Імплементування базових анімацій	3 дні	Фіналізація початкового бачення продукту, Визначення програмного середовища для задачі
2.17.	Розробка дизайну UI	5 днів	-
2.18.	Дизайн ефектів та часток	5 днів	Затвердження технічного завдання
2.19.	Фіналізація візуального дизайну складових ігрового процесу	15 днів	Проєктування плану розробки
2.20.	Дизайн звукового оформлення	5 днів	Проєктування плану розробки
2.21.	Написання композицій	20 днів	Підготовка програм для коммунікацій
2.22.	Фіналізація звукового оформлення	5 днів	Ініціалізація програмного забезпечення для менеджменту проєкту
2.23.	Ведення блогу розробки	45 днів	Ініціалізація програмного забезпечення для менеджменту проєкту, Підготовка ігрового рушія
2.24.	Нарада щодо проведеної роботи з тестової версії	1 день	Ініціалізація розробки
2.25.	Імплеменування змін до пре-альфа версії	10 днів	Створення базових елементів геймплею
2.26.	Завершення розробки альфа-версії	1 день	Розробка мінімального набору ігрових механік

11

Рис. Д3.11

План спринтів

Код	Контрольна віха	Часові рамки
1	Стадія планування	01.02.23 – 14.03.23
2	Розробка гри - спринт 1	15.03.23 – 30.03.23
3	Розробка гри - спринт 2	01.04.23 – 20.04.23
4	Розробка гри - спринт 3	21.04.23 – 10.05.23
5	Розробка гри - спринт 4	11.05.23 – 01.06.23
6	Розробка гри - спринт 5	02.06.23 – 22.06.23
7	Розробка гри - спринт 6	23.06.23 – 13.07.23
8	Стадія контролю якості	24.07.23 – 19.09.23
9	Реліз продукту	20.09.23 – 18.10.23

12

Рис. Д3.12

План управління ризиками

Ризик	Попередження ризику	Усунення ризику
Зрив темпу робіт	Моніторинг виконання проєкту, перепланування затримок	Виділення потужностей на проблемні задачі, залучення зовнвшньох підтримки
Недостатній рівень підготовки персоналу	Якісний відбір кандидатів на посади, професійна переверка їх знань та навичок	Пошук нового персоналу
Збільшення витрат під час виконання	Врахувати додаткові витрати під час планування бюджету, розробити детальний план витрат для проєкту	Мінімізація витрат через перерозподілення грошових ресурсів
Звільнення ключових працівників	Якісний відбір кандидатів на посади, набір	Заміна члена команди, по можливості переведення частини інших працівників на завдання
Недостатнє фінансування	Просування продукту, ведення блогу розробки, залучення платників через спеціальні пропозиції	Залучення кредитних коштів, мінімізація втрат через пререрозподілення коштів
Проблеми комунікації в команді	Чіткий регламент робіт, запровадження професйної етики на робочому місці	Проведення наради по проблемним питанням, усунення конфліктів через технічні фактори, реогранізація команди
Помилки в формулюванні завдань проєкту	Перевірка статуту проєкту і проєктної документації, складання детального плану проєкту, визначення вимог та відповідальностей працівникків	Збільшення часових рамок проєкту та виправлення неточностей
Вибір неоптимальної схеми реалізації розробки продукту	Детальний аналіз можливостей, слабких і сильних сторін кожного з доступних варіантів, використання досвіду індустрії, відкритість обговорення деталей проєкту	Рефоматування методів розробки, перехід на інші фреймворки розробки

12

Рис. Д3.13

Висновки

1. Ігрова індустрія – це досить прибутковий і переспективиний бізнес, на сцені якого Україна представлена досить слабко. Освоєння цього ринку зможе привнести в країну велику кількість іноземних інвестицій та сучасних практик розробки програмного забезпечення.
2. Були досліджені проєкти, схожі на виконуваний. У результаті були встановлені найбільш важливі елементи таких продуктів: цікаве оформлення, якісний музичний супровід, комплексний дизайн рівнів, механічно складна бойова система, яка при цьому не перевантажує гравця.
3. Цілі проєкту розробки “Runic Forest”: розробка процедурного генератора рівнів, впровадження комплексного та захоплюючого ігрового циклу, впровадження оригінального музичного супроводу, імплементація мережевої підтримки для мультплеєрного режиму.
4. Обмеження проєкту: дата завершення роботи над проєктом: 18.10.23, виставлено обмеження загального бюджету: 1 100 000 гривень. Серед обмежень організації робіт: використання рушія Unity, використання ліцензійного програмного забезпечення, мова програмування C#. На виконання проєкту виділено 186 днів та 859 130 гривень. Для резерву було прийняте рішення виділити додаткові кошти, 240 000 гривень.
5. У результаті був комплексно проаналізований процес роботи над продуктом і отриманий чіткий план розробки програмного забезпечення “Runic Forest”.

13

Рис. Д3.14

Дякую за увагу!

Рис. Д3.15