Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет *Комп'ютерних наук*
(повна назва)

Кафедра *Системотехніки*
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

Другий (магістерський)
(рівень вищої освіти)

Розроблення компонентів підсистеми об'ємного моделювання для інформаційних технологій проектування
(тема)

Виконав:
Студент *II* курсу, групи *ІТПм-22-1*

Бутирін М. С.
(прізвище, ініціали)

Спеціальність *122 Комп'ютерні науки*
(код і повна назва спеціальності)

Тип програми *Освітньо-професійна*
(освітньо-професійна або освітньо-наукова)

Освітня програма *Інформаційні технології проектування*
(повна назва освітньої програми)

Керівник *проф. Безкоровайний В. В.*
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри системотехніки _____ *Гребеннік І. В.*
(підпис) (прізвище, ініціали)

2024 р.

Я як студент ХНУРЕ розумію і підтримую політику закладу із академічної доброчесності. Я не надавав і не одержував недозволену допомогу під час підготовки кваліфікаційної роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

05.01.2024 _____ *Бутирін М.С.*

Кваліфікаційна робота оформлена у відповідності до вимог діючих стандартів та методичних вказівок.

Матеріали кваліфікаційної роботи не містять відомостей, що заборонені для опублікування у відкритих виданнях.

Попередній захист проведено.

Керівник кваліфікаційної роботи _____ *В. В. Безкоровайний*

Харківський національний університет радіоелектроніки

Факультет *Комп'ютерних наук*

Кафедра *Системотехніки*

Рівень вищої освіти *другий (магістерський)*

Спеціальність *122 Комп'ютерні науки*

Освітня програма *Інформаційні технології проектування*

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)
«___» _____ 2023 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові *Бутиріну Микиті Сергійовичу*

1. Тема роботи *«Розроблення компонентів підсистеми об'ємного моделювання для інформаційних технологій проектування»*
затверджена наказом по університету від *«20» листопада 2023 р. №13736 Ст.*
2. Термін подання студентом роботи до екзаменаційної комісії *«15» січня 2024 р.*
3. Вихідні дані до роботи *Об'єкт дослідження – інформаційні технології проектування тривимірних об'єктів. Предмет дослідження – об'ємні моделі об'єктів для арт-виконавців. Предмет розробки – компоненти системи об'ємного моделювання для інформаційної технології проектування у вигляді програми з основними функціями створення і перегляду моделей. Технічне забезпечення: IBM-сумісний персональний комп'ютер.*
4. Перелік питань, що потрібно опрацювати в роботі *Вступ. Огляд та аналіз предметної області. Інформаційні системи як об'єкти проєктування. Арт-об'єкти моделювання. Технологія проектування інформаційних систем. Огляд систем-аналогів об'ємного моделювання (ОМ). Обґрунтування мети вирішення поставленої. Технологія проектування системи ОМ. Виявлення вимог та постановка задачі на розробку системи. Вибір методології розробки. Вибір програмного забезпечення. Стратегія розробки компонентів системи та їх функціональний тип. Моделювання потоків даних системи. Планування та організація розробки системи. Реалізація системи об'ємного моделювання. Проектування інтерфейсу користувача. Проектування основної механіки системи. Перспективні напрямки розробки (стратегії та майбутні розширення). Висновки. Перелік джерел посилання.*

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій *кресленики, схеми, плакати та/або комп'ютерні ілюстрації (слайди) на аркушах формату А4, що включаються до тексту пояснювальної записки або складу додатків: схема технології проектування інформаційних систем; діаграми функціонального моделювання системи; діаграми потоків даних системи; екранні форми програми.*

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	*Отримання завдання на кваліфікаційну роботу*	*20.11.2023*	*Виконано*
2	*Огляд сучасних інформаційних технологій*		
	моделювання та проєктування систем	*27.11.2023*	*Виконано*
3	*Постановка задачі на розробку системи*	*30.11.2023*	*Виконано*
4	*Вибір середовища розробки програми*	*27.11.2023*	*Виконано*
5	*Проектування та розробка системи*	*30.11.2023*	*Виконано*
6	*Підготовка публікацій за результатами*		
	дослідження	*01.12.2023*	*Виконано*
7	*Оформлення пояснювальної записки*	*25.12.2023*	*Виконано*
8	*Подання закінченої роботи науковому*		
	керівникові	*28.12.2023*	*Виконано*
9	*Усунення зауважень наукового керівника*	*05.01.2024*	*Виконано*
10	*Підготовка презентації*	*08.01.2024*	*Виконано*
11	*Подання роботи на рецензування*	*10.01.2024*	*Виконано*
12	*Попередній захист*	*12.01.2024*	*Виконано*
13	*Подання роботи до екзаменаційної комісії*	*15.01.2024*	*Виконано*

Дата видачі завдання *«20» листопада 2023* р.

Студент _____ *Бутирін М. С.*
(підпис) (прізвище, ініціали)

Керівник роботи _____ *проф. Безкоровайний В. В.*
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка до магістерської кваліфікаційної роботи: 80 с., 1 табл., 24 рис., 4 дод., 32 джерела.

АРТ-ПРОЄКТ, ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ ПРОЕКТУВАННЯ, ОБ'ЄМНЕ МОДЕЛЮВАННЯ, КАСКАДНА МОДЕЛЬ РОЗРОБКИ, C#, IDEF0, DFD, UML, UNITY 3D.

Об'єкт дослідження – інформаційні технології проектування тривимірних об'єктів.

Предмет дослідження – об'ємні моделі об'єктів для арт-виконавців.

Мета кваліфікаційної роботи – підвищення ефективності інформаційних технологій створення арт-проєктів за рахунок розробки компонентів підсистеми об'ємного моделювання.

Методи дослідження – системний підхід, методи структурного аналізу і моделювання, теорія прийняття рішень, методи сучасних інформаційних технологій.

У роботі на основі рушія Unity 3D реалізовано систему об'ємного моделювання для інформаційних технологій створення арт-проєктів, спроектовано та розроблено інтерфейс користувача, розроблено основну механіку роботи системи.

Галузь застосування – дизайн об'ємних і двомірних арт-моделей у розважальній сфері. Розробка надає можливість користувачам без попередніх навичок працювати з об'єктами у тривимірному просторі для швидкого створення прототипів своїх ідей.

ABSTRACT

Explanatory note to the master's qualification thesis: 80 p., 1 tab., 24 pic., 4 app., 32 sources.

ART PROJECT, DESIGN INFORMATION TECHNOLOGY, VOLUME MODELING, CASCADE DEVELOPMENT MODEL, C#, IDEF0, DFD, UML, UNITY 3D.

The object of research is information technologies for designing three-dimensional objects.

The subject of research is three-dimensional models of objects for art performers.

The goal of the qualification work is to improve the efficiency of information technologies in creating art projects through the development of components of the three-dimensional modeling subsystem.

Research methods – systematic approach, methods of structural analysis and modeling, theory of decision-making, methods of modern information technologies.

In the work based on the Unity 3D engine, a three-dimensional modeling system was implemented for information technologies for creating art projects, the user interface was designed and developed, and the basic mechanics of the system were developed.

The field of application is the design of three-dimensional and two-dimensional art models in the field of entertainment. The development enables users without previous skills to work with objects in three-dimensional space to quickly create prototypes of their ideas.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

Баг – помилка при виконанні програми.

Емуляція – відтворення програмними або апаратними засобами роботи інших програм або пристроїв.

Інтерфейс – сукупність програмних та апаратних засобів, що забезпечують взаємодію користувача та системи.

ПК – персональний комп’ютер.

Сетинг – сукупність різнопланових елементів, які однозначно ідентифікують світ, де відбувається події певної художньої картини.

СОМ – система об’ємного моделювання.

Рендеринг – процес отримання зображення за моделлю за допомогою комп’ютерних програм.

UML – Unified Modeling language, уніфікована мова моделювання, використовується у парадигмі об'єктно-орієнтованого програмування.

ВСТУП

У наш час Інтернет заповнений різноманітними розважальними ресурсами, що надають нам великий вибір можливостей для розваг та творчості. Від книг та ігор до серіалів та фільмів – усе це вже перемістилося до віртуального простору. Проте, слід визнати, що особливу роль у цьому процесі відіграють арт-виконавці, творці, які працюють над створенням вмісту.

У цьому контексті важливо підкреслити необхідність наявності для арт-виконавців інструменту, що надає можливість швидко, ефективно та інтуїтивно створювати та редагувати тривимірні моделі. Це стає невід’ємною частиною їхнього творчого процесу.

Крім того, вирішальну роль відіграє можливість надати доступ до цих інструментів не тільки досвідченим фахівцям, але й початківцям, які тільки розпочинають свій шлях у цій галузі. Важливо, щоб створення моделей не вимагало від них складних технічних навичок чи поглиблених знань у сфері об’ємного моделювання.

Арт-виконавці, що створюють цей контент, мають особливу потребу в відповідних інструментах. Швидкість розвитку технологій у сучасному світі надає нам можливість користуватися все більш потужними та доступними засобами для творчості. Відкривається великий простір для створення нових тривимірних об’єктів.

Важливо врахувати, що програмні системи об’ємного моделювання є важливим компонентом у процесі роботи з тривимірними об’єктами. Такі системи дозволяють арт-виконавцям реалізовувати свої ідеї з високою точністю та деталізацією. Системи об’ємного моделювання є критично важливими у їхній творчій діяльності, надаючи можливість втілювати у життя найскладніші концепції та ідеї з найвищою якістю [1– 3].

Отже, створення інноваційного ресурсу, який надасть можливість дизайнерам або письменникам втілити свої творчі концепції у тривимірному середовищі, є важливим кроком у розвитку цієї сфери.

Мета кваліфікаційної роботи – підвищення ефективності інформаційних технологій створення арт-проєктів за рахунок розробки компонентів підсистеми об'ємного моделювання.

Для досягнення мети необхідно вирішити такі завдання:

– виконати аналіз комп'ютерних інформаційних систем як об'єктів проектування;

– виконати огляд та аналіз сучасних систем об'ємного моделювання арт-об'єктів;

– обґрунтувати вимоги до створюваної систем об'ємного моделювання арт-об'єктів і технологію її проектування;

– здійснити планування та організацію розробки системи;

– розробити інтерфейс користувача та засоби основної механіки системи.

Об'єкт дослідження – інформаційні технології проектування тривимірних об'єктів.

Предмет дослідження – об'ємні моделі об'єктів для арт-виконавців.

Методи дослідження – системний підхід, методи структурного аналізу і моделювання, теорія прийняття рішень, методи сучасних інформаційних технологій.

У роботі на основі рушія Unity 3D реалізовано систему об'ємного моделювання для інформаційних технологій створення арт-проєктів, спроектовано та розроблено інтерфейс користувача, розроблено основну механіку роботи системи.

За результатами дослідження опубліковано тези доповіді в матеріалах міжнародної науково-практичної конференції «Distance learning: problems, ways of development and the latest technologies» (December 25-27 2023, Munich, Germany) [4].

1 ОГЛЯД ТА АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Комп’ютерні інформаційні системи як об’єкти проєктування

В основі інфраструктури сучасних компаній з різних сфер людської діяльності лежать комп’ютерні інформаційні системи. Умови конкуренції вимагають постійного удосконалення існуючих та створення нових інформаційних систем, що обумовлюється, зокрема [5]:

– завершенням періодів автоматизації існуючих підрозділів і їх функцій;

– необхідністю моделювання і створення нових бізнес-процесів;

– появою нового методологічного забезпечення щодо проектування інформаційних систем, включаючи управління вимогами;

– швидким розвитком технологій Інтернет, що надають якісні і своєчасні послуги з поставки і передачі інформації.

Специфіка технологій створення і розвитку інформаційних технологій полягає у тому, що питання аналізу предметної області, розробки й аналізу вимог до інформаційних систем, побудови їх комплексних моделей вирішуються на ранніх етапах аналізу та системного проектування.

Задачі наступних етапів розв’язуються існуючими засобами сучасних інформаційних технологій.

Накопичення досвіду розробниками сучасних інформаційних систем здійснюється у двох основних напрямках:

– створення типових інструментальних засобів розробки інформаційних систем;

– розробки і впровадження унікальних інформаційних систем та інформаційних технологій.

Тиражування типових проєктних рішень без урахування специфіки компанії чи технологій, які використовуються нею, у загальному випадку не

влаштовує замовників. Розробка унікальних інформаційних технологій для окремої компанії вимагає значних фінансових і часових витрат.

Виходом є модернізація, розвиток чи реінжиніринг існуючих комп'ютерних інформаційних систем (КІС) компанії. Для цього необхідно удосконалювати існуючі та створювати нові інструменти для удосконалення існуючих інформаційних систем і технологій. З цією метою використовую два основні підходи [5]:

– лобістський підхід;

– підхід на основі автоматизації існуючого "хаосу".

Лобістський підхід передбачає формальний аналіз існуючих структури і системи управління фірмою розробником комп'ютерних інформаційних технологій з подальшою їх адаптацією функціональних комплексів інформаційних технологій (ФКІТ) типу MRP, ERP.

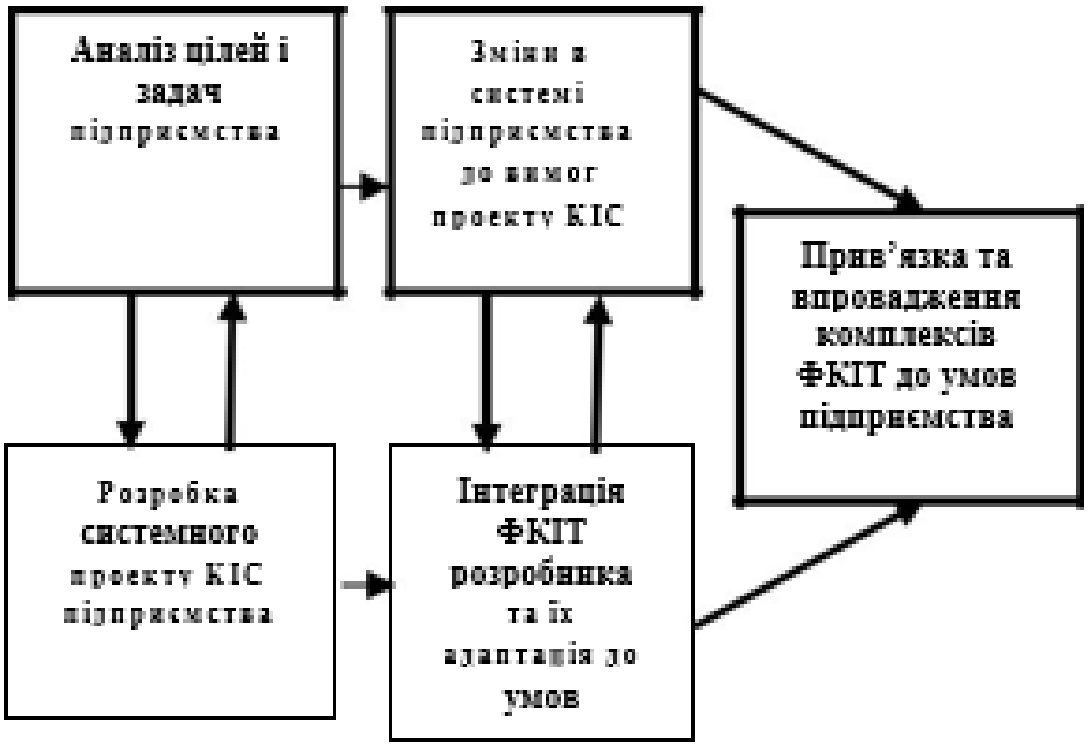

Рисунок 1.1 – Схема створення ІТ підприємства із застосуванням «лобістського» підходу [5]

При використанні підходу на основі автоматизації існуючого «хаосу» на першому етапі фірма-розробник визначає існуючі структури компанії (схеми управління, технології, потоки документів тощо). На другому етапі їх шляхом реалізують у власній інформаційній технології, адаптуючи її до умов компанії впровадження.

Рисунок 1.2 – Схема на основі автоматизації існуючого «хаосу» [5]

Розглянуті підходи часто не задовольняють замовників, які бажають не стільки автоматизувати процеси управління, скільки одержати реальну користь від функціонування компанії в цілому.

Одним з перспективних підходів є підключення керівництвом компаній незалежних спеціалістів-консультантів для системного аналізу і розробки системного проекту перспективної інформаційної технології.

Процес створення системи об’ємного моделювання для технології проєктування як інформаційної системи подається за допомогою таких понять: життєвий цикл, фаза, стадія, етап, робота. Послідовність робіт зі створення системи об’ємного моделювання подається у вигляді процесу, що має ієрархічний опис.

Схема декомпозиції проблеми розробки системи об’ємного моделювання (СОМ) подається у вигляді комплексу підпроблем *SubProblem* і окремих задач *Task* різних рівнів зі зв'язками за вхідними

даними і результатами розв'язання [4]:

$$Problem = \{ SubProblem^l \},\ SubProblem^l = \{ Task_i^l \},\ l = \overline{1,n_l},\ i = \overline{1,i_l}, \quad (1.1)$$

де n_l – кількість рівнів декомпозиції;

$Task_i^l$ – i-та задача l-го рівня;

i_l – кількість задач на l-му рівні.

До першого рівня декомпозиції відноситься комплекс таких підпроблем: $SubProblem_1$ – формування вимог до системи об'ємного моделювання та розробка технічного завдання на її створення; $SubProblem_2$ – розробка СОМ; $SubProblem_3$ – адаптація (модернізація) СОМ; $SubProblem_4$ – реінжиніринг СОМ.

На нижньому рівні декомпозиції виділяється множина елементарних робіт, що розглядаються як окремі локальні задачі $\{ Task_i \}$, $i = \overline{1,n}$. Такі задачі циклічно розв'язуються, з урахуванням вимог до СОМ та показників її якості:

– необхідний функціонал системи;

– вимоги до використовуваних технічних засобів (бажана частота процесора, необхідні об'єми пам'яті);

– рівень необхідних знань користувача;

– вартість тощо.

Подібні вимоги можуть розглядаються як обмеження або локальні критерії ефективності $k_j(s)$, $j = \overline{1,m}$ (де s – варіант побудови СОМ; m – кількість локальних критеріїв).

Існуючі технології підтримки прийняття проєктних рішень ґрунтуються на ранжуванні допустимих варіантів $S = \{ s \}$ з використанням кардиналістичного та ординалістичного підходів [5–10].

У рамках кардиналістичного підходу переваги між показниками

якості систем об'ємного моделювання можуть бути задані значеннями вагових коефіцієнтів λ_j локальних критеріїв $k_j(s)$, $j=\overline{1,m}$. За наявності локальних критеріїв, що мають різну направленість $k_i(s)\to min$ s $k_j(s)\to max$ використовують функції їх корисності $\xi_j(s)$, $j=\overline{1,m}$:

$$\xi_j(s)=k_j(s)-k_j^-)/(k_j^+-k_j^-), \tag{1.2}$$

де $k_j(s)$ – значення j-го локального критерію для s-го варіанта побудови СОМ, $j=\overline{1,m}$;

k_j^+ – найкраще значення j-го локального критерію;

k_j^- – найгірше значення j-го локального критерію.

Найкращому варіанту побудови СОМ відповідатиме максимум адитивної функції загальної корисності:

$$s^o=arg\ \max_{s\in S}\left\{P(s)=\sum_{j=1}^{m}\lambda_j\xi_j(s)\right\}. \tag{1.3}$$

При цьому вагові коефіцієнти локальних критеріїв $\lambda_j\geq 0$, $j=\overline{1,m}$ повинні задовольняти умові нормування $\sum_{j=1}^{m}\lambda_j=1$.

У ситуації, коли чисельні значення вагових коефіцієнтів λ_j не можуть бути визначені кількісно, локальні критерії пропонується впорядковати за ступенем їх важливості $k_1\succ k_2\succ ...\succ k_m$.

Тоді для вибору найкращого варіанту СОМ системи пропонується використати метод лексикографічної оптимізації. Варіант побудови СОМ s є лексикографічно кращим ніж варіант v, якщо виконується одна з умов:

$$\xi_1(s)>\xi_1(v);$$

$$\xi_1(s)=\xi_1(v),\ \xi_2(s)>\xi_2(v);$$

$$\dots$$

$$\xi_j(s)=\xi_j(v),\ j=\overline{m-1},\ \xi_m(s)>\xi_m(v).$$

На першому етапі для вибору найкращого варіанта за цим методом на множині допустимих варіантів побудови систем об'ємного моделювання $S=\{s\}$ необхідно знайти підмножину варіантів $S_1^o \subseteq S$, оптимальних за локальним критерієм $k_1(s)$.

На другому етапі необхідно знайти підмножину варіантів $S_2^o \subseteq S_1^o$, оптимальних за локальним критерієм $k_2(s)$ і так далі.

На останньому етапі із підмножини варіантів $S_{m-1}^o \subseteq S_{m-2}^o$, оптимальних за критерієм $k_{m-1}(s)$, за критерієм $k_m(s)$ обирається найкращий варіант побудови СОМ $s^o \in S_{m-1}^o$.

При чому, якщо при виборі за локальними критеріями $k_j(s)$, $j=\overline{m-1}$ буде обрано лише один варіант, відповідну підмножину S_j^o, $j=\overline{m-1}$ необхідно розширити за рахунок квазіоптимальних варіантів побудови СОМ.

Слід враховувати, що занадто малий розмір поступки (відхилення від найкращого значення) не дозволяє враховувати значення всіх локальних критеріїв якості системи об'ємного моделювання, а занадто великий – врахувати задане упорядкування критеріїв.

В сучасних інформаційних технологіях, що використовуються в різних сферах людської діяльності (проєктування, управління, освіта, мистецтво), все більш широке застосування знаходять засоби 3D-моделювання [1–4]. Більшість сучасних систем 3D-моделювання є надто складними для освоєння та ефективного застосування користувачами, зокрема, в арт-галузі, що не є IT- фахівцями [11–17].

1.2 Задачі 3D-моделювання в соціокультурній сфері

Розглянемо задачі 3D-моделювання та проектування для арт-виконавців [16–17].

Сучасний прогрес у цій галузі вимагає вдосконалених інструментів 3D-моделювання. Велика кількість людей з усього світу захоплюється фантастичним сетингом, і це призводить до зростання потреби у спеціалізованих програмах, які дозволять опрацьовувати та реалізовувати нові ідеї.

Цей феномен призводить до зростання потреби у спеціалізованих засобах, зокрема програмному забезпеченні, яке надасть можливість арт-виконавцям реалізувати свої новаторські ідеї. Наприклад, у сучасний період спостерігається стабільне зростання використання розважальних ресурсів, спрямованих на фантастичний сетинг майбутнього.

Зокрема, арт-виконавцям необхідно джерело натхнення, яке б стимулювало їх творчий потенціал. 3D-дизайнерам, у свою чергу, необхідні готові моделі для втілення своїх ідей в сценах. Письменникам необхідні ілюстрації для своїх повістей. Дизайнерам – генерація ідей. Практично всі, хто беруть участь у створенні мистецьких робіт, потребують ефективних інструментів для швидкого створення зображень що втілюють всесвіт.

Сучасний розвиток технологій у галузі комп’ютерної графіки створює унікальні можливості для арт-виконавців. Вони потребують найефективніших інструментів для створення та редагування тривимірних об’єктів, що дозволяє їм реалізовувати свої творчі задуми з високою точністю та деталізацією.

Також важливо врахувати, що існують різноманітні рішення для проектування 3D-моделей, проте багато з них вимагають значних затрат часу та коштів. Наприклад, наймання 3D-дизайнера чи придбання готових моделей у Інтернеті. Ці варіанти можуть бути виправданими для великих

команд чи проектів, але не завжди є економічно доцільними для авторів, які працюють самостійно.

1.3 Арт-об’єкти моделювання

Об’єктовим моделювання може бути будь-який тривимірний об’єкт. Об’єкт, що досліджується, є тривимірним віртуальним арт-елементом, що визначається його геометричною структурою, текстурою та розташуванням у тривимірному просторі. Цей арт-об’єкт втілює творчі концепції арт-виконавців та виступає у ролі джерела натхнення для творчих проектів [16-17].

Геометрична структура цього об’єкта визначає його форму, розміри та розташування компонентів. Текстура, яку він має, надає йому характер та відтворює естетичні аспекти творчого вираження. Розташування у тривимірному просторі визначає його позицію та орієнтацію, що є важливим для віртуальної взаємодії з ним.

Арт-об’єкт є не лише як візуальний елемент, але й як інструмент для розвитку та вираження творчих ідей. Він є частиною сучасної технічної системи для моделювання та дизайну, і його аналіз дозволяє розуміти, як технології впливають на творчий процес арт-виконавців та їхню здатність реалізовувати свої творчі задуми.

1.4 Технологія проектування інформаційних систем

Технології проектування інформаційних систем представляють собою комплексний набір інструментів та методів, спрямованих на розробку та впровадження ефективних інформаційних рішень. Ці технології включають в себе різноманітні підходи до аналізу, проектування та управління інформаційними системами для вирішення конкретних завдань та потреб організацій [17].

Приклад схеми технології проектування інформаційних систем подано на рисунку 1.3.

Рисунок 1.3 – Схема технології проектування інформаційних систем [17]

Один із основних аспектів технологій проектування інформаційних систем – це визначення вимог до системи. Цей етап включає в себе вивчення потреб користувачів та бізнес-вимог для визначення функціональних та нефункціональних характеристик системи. При цьому можуть використовуватися методи аналізу бізнес-процесів, вивчення вимог до даних та інші техніки.

Далі, технології проектування включають розробку архітектури інформаційної системи. Цей етап передбачає визначення структури, компонентів та взаємозв’язків системи для забезпечення її ефективної роботи. Моделювання, діаграми потоків та архітектурні патерни можуть використовуватися для візуалізації та документування цього процесу.

Однією з ключових складових технологій проектування інформаційних систем є вибір технологій розробки та програмного забезпечення. Врахування вимог проекту, вибір оптимальних мов

програмування, фреймворків та інших інструментів визначає успішність реалізації системи.

Крім того, технології проектування інформаційних систем включають етап тестування, документування, впровадження та подальшого супроводу системи. Використання сучасних засобів автоматизації та методів контролю якості гарантує успішне впровадження та функціонування інформаційних систем.

1.5 Огляд систем-аналогів для об’ємного моделювання

Існує низка систем моделювання, проте більшість з них виявляються надто складними для ефективного освоєння та використання. У багатьох випадках, вигідніше придбати готову модель чи замовити її створення на замовлення, ніж витрачати час на самостійне оволодіння методами проектування.

Середовища розробки 3D-моделей, такі як 3ds Max, Maya, Blender та інші, є потужними інструментами для створення та редагування тривимірних об’єктів. Вони надають великі можливості, проте вимагають глибоких знань та досвіду в роботі з 3D моделюванням. Ці методи можуть бути через мірно дорогими як для особистого використання, так і для замовлення моделей від дизайнерів [18–20].

Інженерні системи моделювання, такі як Renga, LibreCAD та інші, є ще складнішими в розробці об’ємних об’єктів, але вони мають конкретно визначені налаштування та функції, адаптовані до конкретних галузей використання. Поріг вхідних знань для роботи з такими системами є високим [21–22].

Системи з використанням штучного інтелекту, такі як Artbreede, є потужними інструментами, проте вони мають свої обмеження та недоліки. Вони дозволяють користувачам завантажити два зображення та отримати їх комбінований варіант. Використання таких систем інтуїтивно зрозуміле та

просте, проте створення та навчання штучного інтелекту виявляється важким завданням. Безпосереднє використання отриманих моделей для ліцензійних цілей неможливе [15].

Системи штучної 3D-графіки, такі як Adobe та інші, дозволяють створювати прототипи об’ємних моделей у двовимірному середовищі. Вони емулюють об’ємне середовище та дозволяють малювати тривимірні об’єкти у двовимірному просторі. Результат залежить від малюнка, тому важливе вміння малювати [23].

Внутрішні ігрові конструктори, такі як Terra Tech, Kerbal Space Program та інші, надають можливість моделювати тривимірні об’єкти в ігровому середовищі. Вони інтуїтивно зрозумілі та придатні для користувачів різного рівня досвіду у роботі з 3D-редакторами. Проте їхні можливості обмежені відповідно до потреб гри, та моделі, створені в них, неможливо використовувати в інших середовищах [24–25].

1.6 Обґрунтування мети вирішення поставленої задачі

В сучасному світі арт-виконавцям дуже потрібні ефективні методи швидкого створення прототипів об’єктів, проте наразі доступні на ринку рішення не відповідають цим потребам. Саме тому важливо розробити нове середовище, яке задовольнить вимоги арт-виконавців. Це середовище повинно надати можливість працювати з об’єктами у тривимірному просторі навіть користувачам без попередніх навичок. Воно також повинно бути відкритим для тих, хто вже володіє деяким досвідом у роботі з подібними інструментами, але хоче знайти нові шляхи для швидкого створення прототипів своїх ідей. Таке середовище може бути корисним і для фанатів ігор, які вже працювали з ігровими конструкторами, але не мають можливості переносити свої розробки в інші середовища. Крім того, воно може стати важливим інструментом для тих, хто має власну ідею, але не знає, як її реалізувати.

2 ТЕХНОЛОГІЯ ПРОЕКТУВАННЯ СИСТЕМИ ОБ'ЄМНОГО МОДЕЛЮВАННЯ

2.1 Види систем моделювання

2.1.1 Середовище розробки 3D-моделей «Blender»

Система моделювання «Blender» представляє собою високопродуктивний інструмент для розробки тривимірних об'єктів. Наразі розглянемо ключові аспекти цієї платформи:

– повна безкоштовність: Blender доступний для використання абсолютно безкоштовно, що робить його дуже привабливим для широкого кола користувачів;

– відкритий вихідний код: Blender має відкритий вихідний код, що означає, що користувачі можуть вносити власні зміни та вдосконалення до програми відповідно до своїх потреб;

– постійний та швидкий розвиток: Спільнота розробників Blender активно працює над вдосконаленням програми, включаючи усунення багів та додавання нових функцій;

– кросплатформність: Blender підтримується на різних операційних системах, включаючи Windows, macOS та Linux, що робить його доступним для багатьох користувачів;

– найшвидший алгоритм рендерінгу: Blender володіє високопродуктивним алгоритмом рендерінгу, що дозволяє швидко створювати візуалізації тривимірних об'єктів.

Для того щоб ефективно використовувати Blender, необхідно мати базові навички роботи з тривимірними об'єктами. Проте навчитися створювати прості моделі та вносити в них зміни можна досить швидко та легко. Blender також підтримує імпорт та експорт моделей, що дозволяє обмінюватися моделями з іншими програмами та платформами.

Приклад інтерфейсу подано на рисунку 2.1.

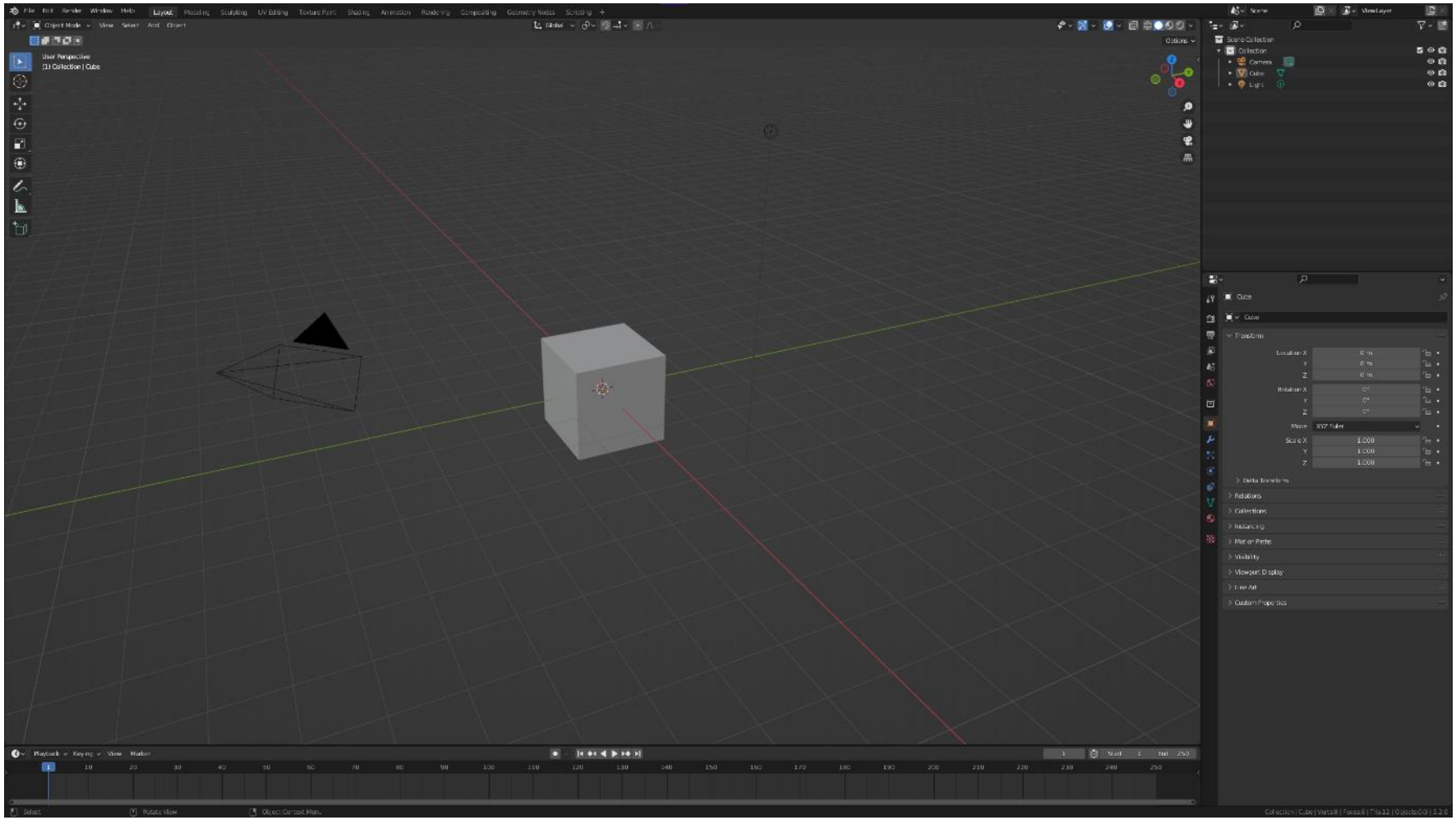

Рисунок 2.1 – Інтерфейс середовища розробки 3D-моделей Blender [20]

2.1.2 Інженерні системи моделювання

Інженерна система моделювання «Renga» є важливим інструментом в галузі проектування об’єктів і відрізняється від своїх аналогів за рядом суттєвих переваг [21]:

– можливість впровадження технології інформаційного моделювання будівель: «Renga» надає можливість ефективно впроваджувати та використовувати інформаційне моделювання будівель, що сприяє збільшенню продуктивності та удосконаленню процесів проектування та будівництва;

– розширене графічне середовище: Платформа «Renga» відзначається наявністю потужного та інтуїтивно зрозумілого графічного інтерфейсу, що дозволяє користувачам здійснювати ефективне проектування та редагування тривимірних моделей;

– можливість розрахунку динамічних впливів: Система надає можливість враховувати динамічні впливи, що є важливим фактором у процесі проектування споруд, що піддаються динамічному навантаженню;

– розгалужена бібліотека кінцевих елементів: «Renga» володіє великою бібліотекою готових кінцевих елементів, що спрощує процес проектування та редагування об’єктів;

– наявність спеціальних елементів: Платформа надає можливість використовувати спеціальні елементи, що спрощує моделювання складних конструкцій та об’єктів.

Для ефективного використання «Renga» необхідно мати високий рівень навичок у створенні та редагуванні об’ємних об’єктів у тривимірному просторі.

Приклад інтерфейсу подано на рисунку 2.2.

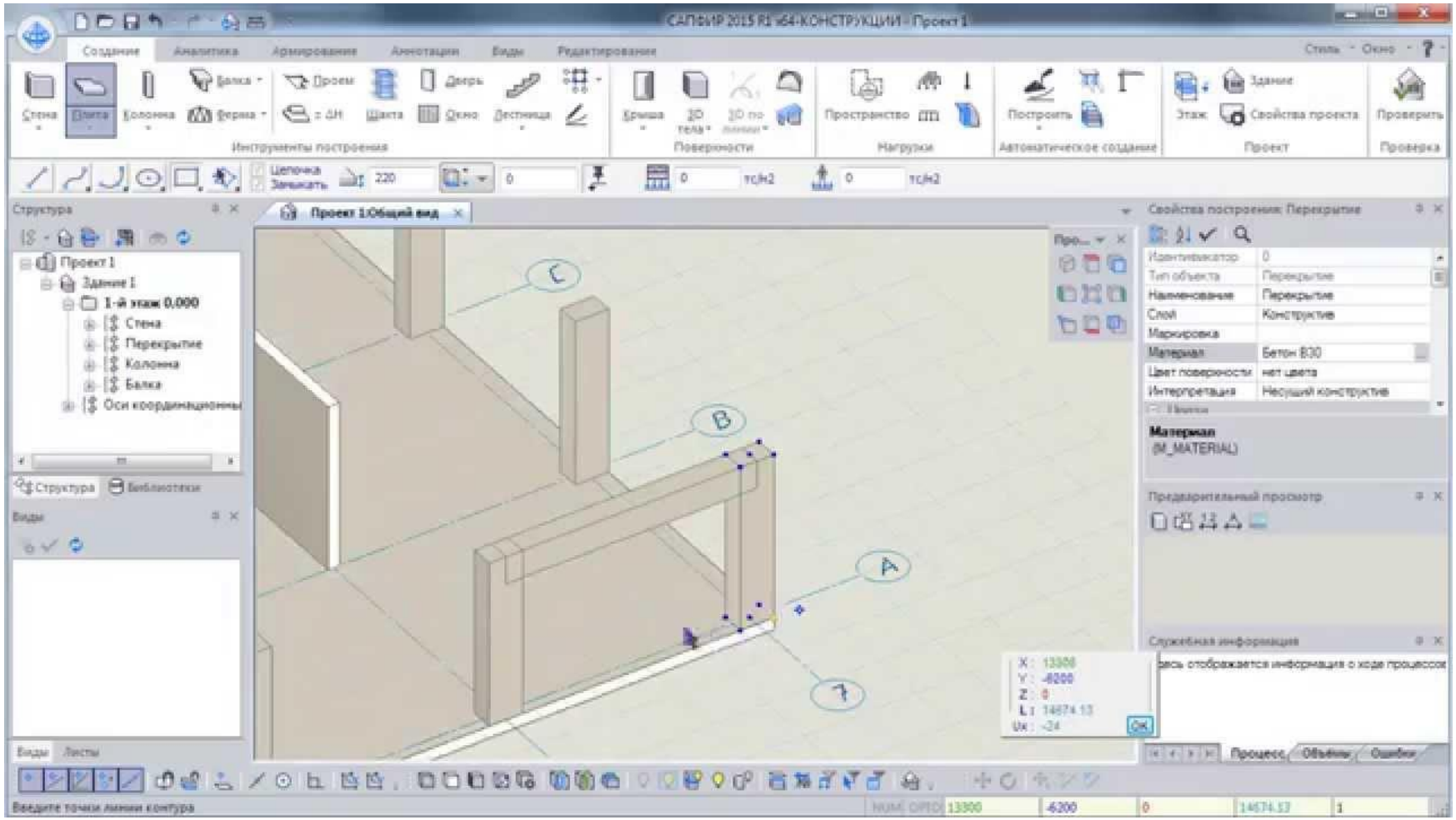

Рисунок 2.2 – Інтерфейс інженерної системи моделювання «Renga» [21]

2.1.3 Системи з основою із штучного інтелекту

Система на основі штучного інтелекту, яка є предметом нашого докладного аналізу – «Artbreede», представляє собою інноваційний підхід до генерації об’ємних моделей [15]. Відмінністю цієї системи є відсутність прямого моделювання 3D-об’єктів в традиційному розумінні. Замість цього вона використовує алгоритми штучного інтелекту для синтезування візуальних репрезентацій з двох вхідних зображень з спільними аспектами. Приклад інтерфейсу подано на рисунку 2.3. Серед ключових переваг варто відзначити наступне:

– відсутність необхідності навчання: Система «Artbreede» вражає можливістю негайного використання без попереднього навчання користувача, що забезпечує оперативність та ефективність у процесі створення перших імітацій об’ємних моделей;

– загальний час створення першої моделі: Застосунок дозволяє генерувати тривимірні репрезентації значно швидше порівняно з традиційними методами моделювання, сприяючи високій продуктивності творчого процесу.

Проте, варто врахувати декілька критичних недоліків:

– необхідність тривалого та глибокого навчання системи для створення нового типу об’єкту: Оскільки створення деяких типів об’єктів може вимагати значних зусиль та часу для навчання системи, цей процес може займати через мірну кількість часу;

– обмеженість можливостей уникнення запозичення елементів з вхідних зображень: У деяких сценаріях система може використовувати деякі елементи з вхідних зображень, що може бути критичним для випадків, де дотримання абсолютної унікальності є критичним фактором.

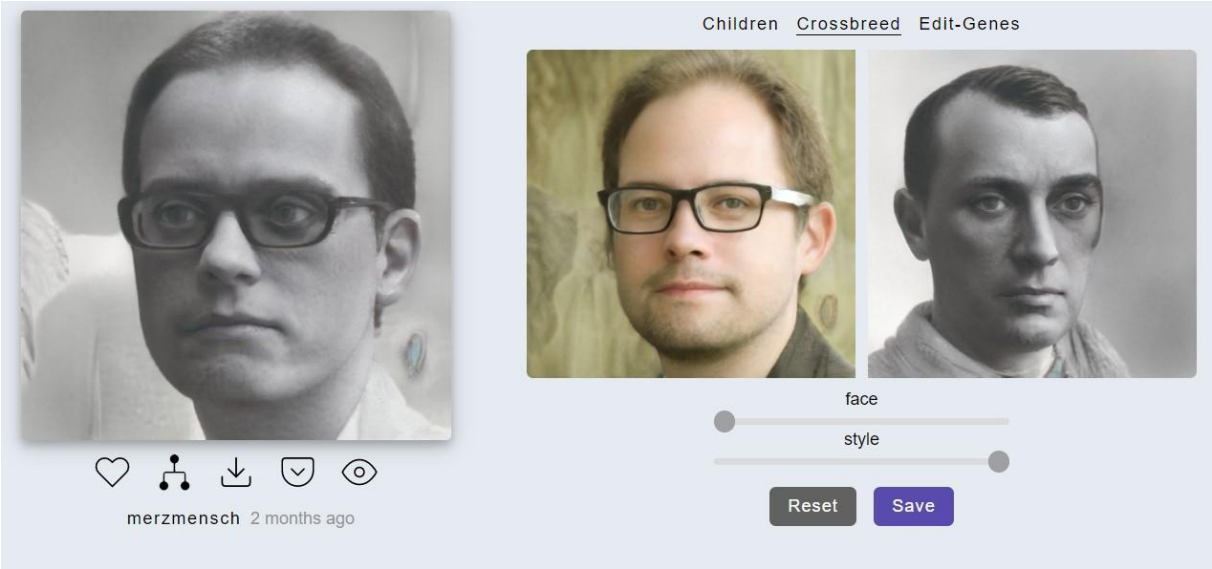

Рисунок 2.3 – Інтерфейс системи з основою із штучного інтелекту Artbreede [15]

2.1.4 Внутрішні ігрові конструктори

Внутрішні ігрові конструктори представляють собою інтегровані в ігрове середовище примітивні системи моделювання, які призначені для виконання певних завдань у межах гри. Для аналізу даного типу систем був обраний ігровий конструктор «Kerbal Space Program», спрямований на побудову та запуск моделей ракет у космос [25]. У цьому конструкторі, моделювання реалізоване у вигляді тривимірної мозаїки, де кожен «шматочок» мозаїки відображає окрему частину космічного корабля. Зразок інтерфейсу ігрового конструктора наведено на рисунку 2.4.

Основною перевагою внутрішніх ігрових конструкторів є їх простота в використанні та швидкість моделювання. Крім того, користувачі можуть тестувати свої моделі безпосередньо у грі. Однак важливими недоліками цих конструкторів є обмеженість можливостей експорту та імпорту об’єктів, а також обмежений функціонал, що час від часу може обмежити можливості користувачів у повному розгортанні їх творчого потенціалу при моделюванні.

Рисунок 2.4 – Інтерфейс ігрового конструктора «Kerbal Space Program» [25]

2.1.5 Порівняння систем 3D-моделювання

Розглянуті системи моделювання пропонують різні підходи та можливості для створення тривимірних об’єктів. Blender відзначається повністю безкоштовним та відкритим вихідним кодом, що робить його потужним інструментом для редагування 3D-моделей. Renga спеціалізується на технології інформаційного моделювання та має розвинуту бібліотеку кінцевих елементів. Artbreede використовує штучний інтелект для імітації об’ємності моделі та не вимагає навчання, проте важко навчити його створювати повністю нові типи об’єктів. Внутрішні ігрові конструктори надають можливість швидко створити прототипи моделей у вигляді ігрових об’єктів, проте їхні можливості щодо експорту та імпорту обмежені. Кожна з цих систем має свої переваги та обмеження.

Результати аналізу подані у таблиці 2.1.

Таблиця 2.1 – Критичні показники розглянутих систем

Показник	Середовище розробки 3D- моделей Blender	Інженерні системи моделювання Renga	Системи з основою зі штучного інтелекту Artbreede	Внутрішні ігрові конструктори Kerbal Space Program
1	2	3	4	5
Безкоштов-ність	Так	Ні	Так	Ні
Рівень необхідних знань	Початковий	Високий	Нульовий	Нульовий
Простота створення	Середньо	Складно	Дуже легко	Легко
Швидкість створення	Середньо	Довго	Дуже швидко	Швидко

Продовження табл. 2.1

1	2	3	4	5
Можливості середовища (переваги над іншими)	Повний контроль моделі, майже нескінченні інструмент, імпорт, експорт,	Повний контроль моделі, майже нескінченні інструменти, імпорт, експорт,	Створення аналогів вже існуючих об’єктів	Можливість протестувати модель
Недоліки середовища (перед іншими)	Потребує вивчення конкретної своєї специфікації	Потребує вивчення конкретної своєї специфікації	Створення аналогів вже існуючих об’єктів (порушення ліцензійного використання, потребує навчання для кожного виду об’єкту	Неможливість імпорту та експорту, обмежений функціонал

2.2 Види систем програмування

У сучасному світі програмування існує множина систем та інструментів, які надають можливість розробки та створення програмного забезпечення. Ці системи відрізняються параметрами, проте основним аспектом, що об’єднує їх, є необхідність оволодіння мовами програмування.

Існує безліч методів та підходів до розробки функціональних програм, які можуть задовольнити наші потреби. У подальшому опису буде надано альтернативи та розглянуто їх особливості.

2.2.1 Рушій Unity 3D

Unity 3D [1] – це середовище для розробки ігор, воно дозволяє розробляти програми на багатьох платформах. Редактор Unity має простий та інтуїтивно зрозумілий інтерфейс що складається з різних видів вікон і на яких показано елементи розробки. За допомогою них можливо одночасно працювати з різними елементами середовища. Інтерфейс системи надано на рисунку 2.5.

Рушій для написання скриптів використовує мови програмування C# та C++, розрахунок фізики проводиться за допомогою рушіїв PhysX та Box2D. Проект Unity поділяється на сцени, що містять свої об’єкти, моделі та налаштування. У системі Unity підтримується система успадкування об’єктів, дочірні елементи успадковують всі зміни положення батьківського елементу.

Unity підтримує стиснення текстур, міпмапінг і різні налаштування роздільності екрана для кожної платформи, забезпечує бамп-мапінг, мапінг відображень, паралакс-мапінг, затінення навколишнього світла у екранному просторі, динамічні тіні за картами тіней, рендер у текстуру та повноекранні ефекти обробки зображення, такі як зернистість, глибина чіткості, розмиття в русі, відблиски віртуальних лінз або ореол навколо джерел світла [1].

Здебільшого за для розробки програм, рушій надає велику кількість інструментів розробки, завдяки яким можливо розробити програму. У Unity є дві великі переваги: наявність візуальної середовище розробки та кросплатформності. Також сильною перевагою є модульність системи компонентів Unity, за допомогою чого відбувається розробка програм. Такий підхід дуже сильно полегшує та прискорює роботу розробників.

Але у такій систем також є і недоліки, а саме: складність роботи у графічному середовищі під час роботи з великою кількістю елементів. Відсутність посилань на зовнішні бібліотеки дуже сильно сповільнює роботу у команді.

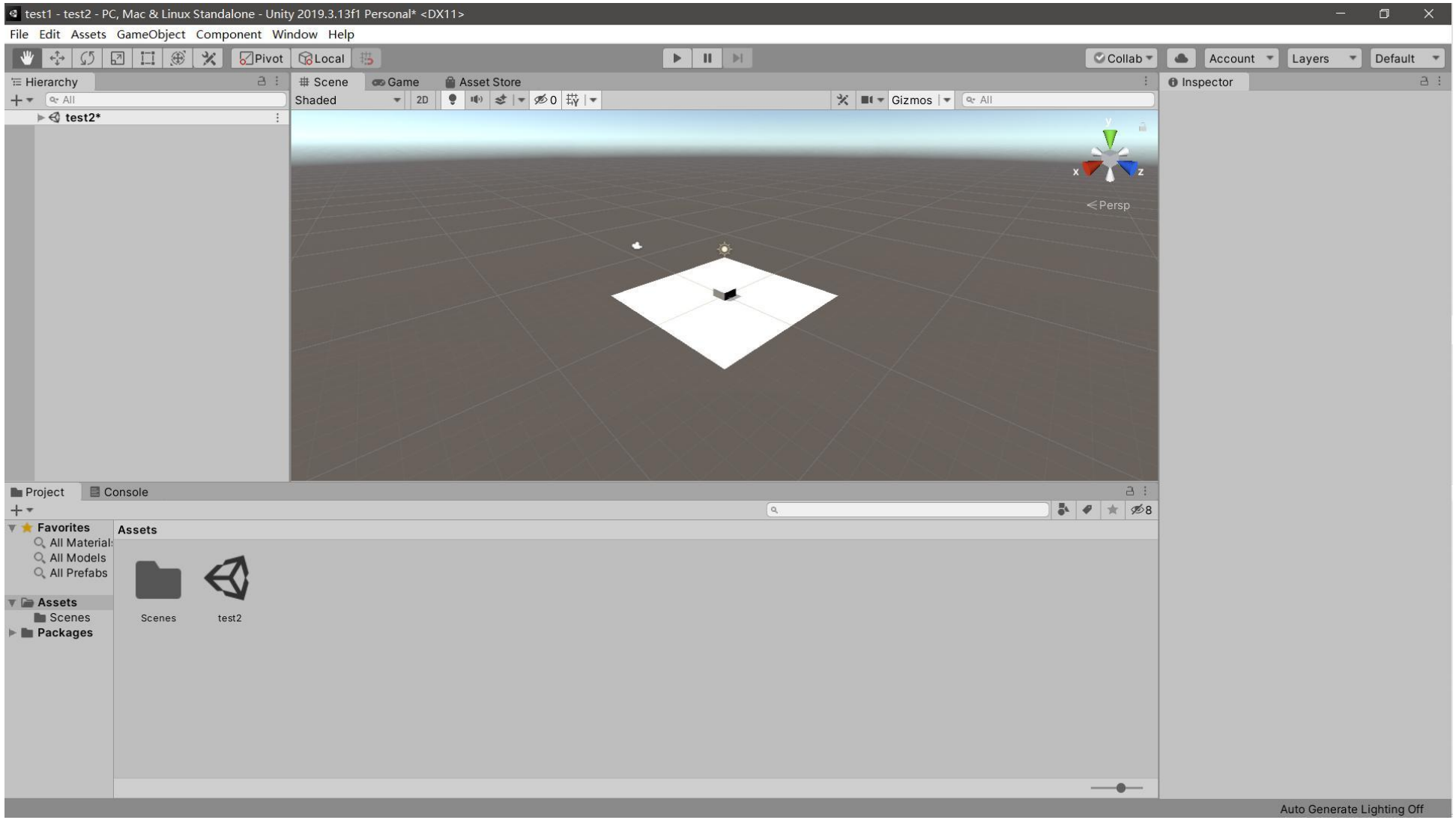

Рисунок 2.5 – Інтерфейс ігрового рушію Unity [1]

2.2.2 Рушій Unreal Engine

Unreal Engine [2] – це універсальне середовище для розробки ігор різноманітних жанрів, що дозволяє розробляти програми на багатьох платформах у тому числі мобільних.

Редактор Unreal Engine виконано у виді багатьох вікон на яких показано елементи розробки. Це забезпечує простий та інтуїтивно зрозумілий інтерфейс.

За допомогою вікон можливо одночасно працювати з різними елементами середовища.

Інтерфейс системи надано на рисунку 2.6.

Рушій написаний на мові C++. Усі елементи рушію подані у вигляді об’єктів з характеристиками та класом, що визначає функціональність об’єкта.

Основні класи налічують: Актор, пішак, світ.

Основний простір розбивається на заповнений та пустий, цей метод дозволяє менше навантажувати систему під час відрисовки сцени.

Для спрощення портування рушій використовує модульну систему залежних компонентів: підтримує різні системи рендерингу (Direct3D, OpenGL, Pixomatic), відтворення звуку (EAX, OpenAL, DirectSound3D), засоби голосового відтворення тексту, розпізнавання мовлення, модулі для роботи з мережею й підтримку різних пристроїв вводу.

Для гри у мережі підтримуються технології Windows Live, Xbox Live, і GameSpy, що дає можливість підключити до 64 гравців (клієнтів) одночасно.

Попри те, що офіційно засоби розробки не містять у собі підтримки великої кількості клієнтів на одному сервері, рушій використовувався для створення MMORPG-ігор [2].

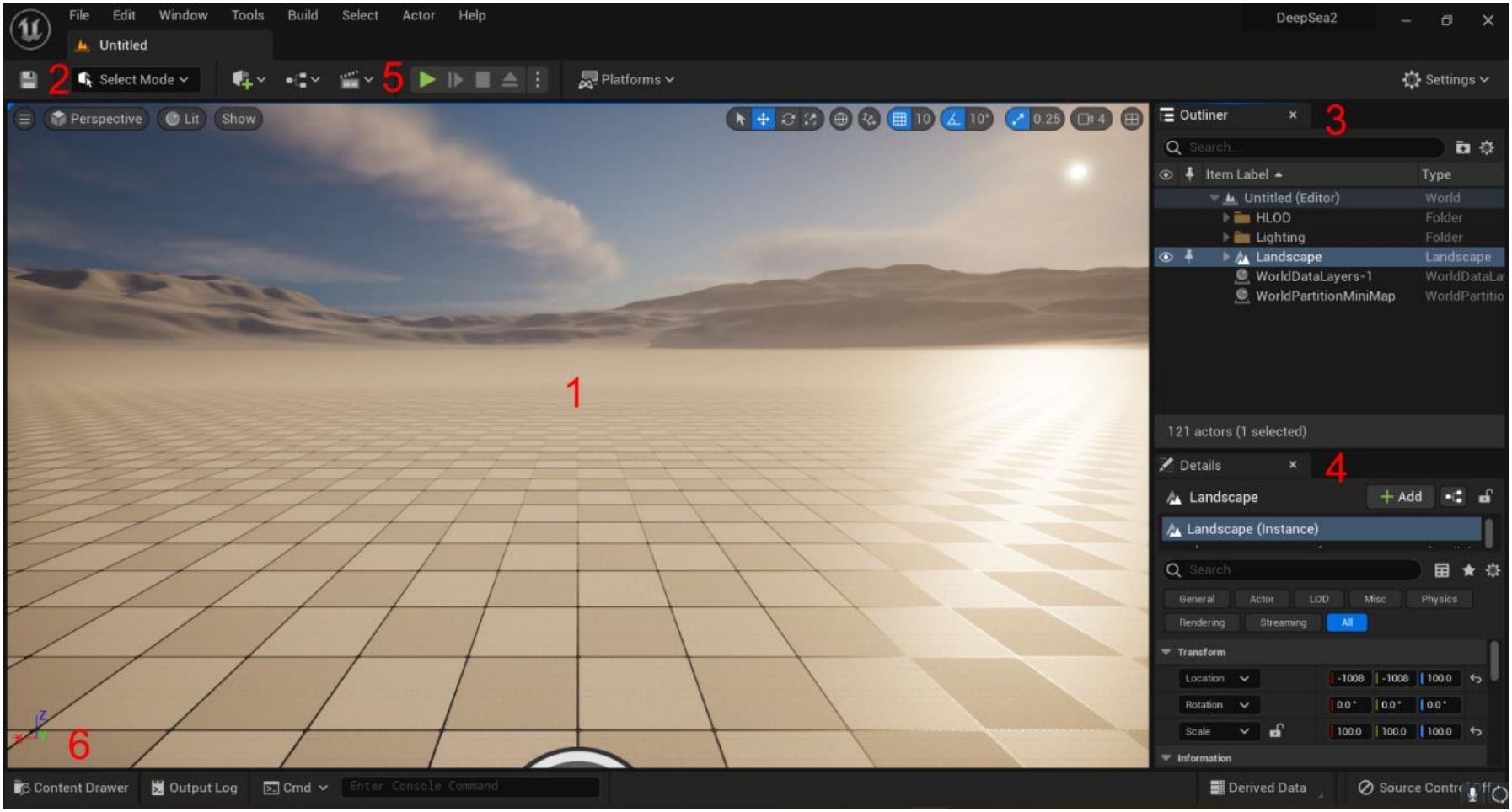

Рисунок 2.6 – Інтерфейс ігрового рушію Unreal Engine [2]

2.2.3 Розробка власного рушію

Розробка власного рушію є складним та затратним процесом, що вимагає великих трудових та фінансових зусиль. Розробка власного рушію може бути складнішою ніж розробка самого застосунку. Цей процес охоплює як програмну реалізацію, так і аналітичну складову, включаючи розробку документації [26].

Хоча розробка власного рушію надає численні переваги, включаючи гнучкість функціоналу та можливість його повного налаштування під конкретний проект, вона супроводжується великими труднощами та витратами. Розробникам рекомендується віддати перевагу готовим рушіям через їх більш доступну вартість та відсутність потреби в постійній технічній підтримці.

Варто зазначити, що великі компанії зазвичай обирають шлях модернізації наявних рушіїв для задоволення власних потреб. Цей підхід дозволяє використовувати вже наявну інфраструктуру та технології, мінімізуючи витрати та збільшуючи швидкість розробки. Важливо відзначити, що використання готового рушія може призвести до обмежень у функціоналі та гнучкості, проте це компроміс заради швидшої та ефективнішої реалізації проекту.

Розробка власного рушію відкриває безліч можливостей для створення унікальних та інноваційних продуктів. Вона дозволяє повністю контролювати технологічний стек, а також реалізовувати найскладніші та найамбіційніші ідеї.

Проте, цей підхід вимагає значних інвестицій, часу та енергії, тому його вибір потребує обдуманих стратегічних рішень. Крім того, у сучасному швидкозмінному технологічному середовищі, часом важливо швидко випустити продукт на ринок, а використання готового рушія може бути більш ефективним у цьому випадку.

2.3 Методології розробки інформаційних систем

Методології розробки – це набір принципів, підходів, правил та процедур, які визначають спосіб організації та управління процесом розробки програмного забезпечення [27].

Методології розробки програмного забезпечення включають у себе різноманітні підходи та набори принципів, спрямованих на організацію та управління процесом розробки програмного продукту. Кожна методологія має свої особливості, переваги та недоліки, і вибір конкретної залежить від специфіки проекту. Надзвичайно важливо обрати ту, яка найбільше відповідає конкретним потребам та метам команди розробників.

Методологія каскадної розробки є однією з класичних та старіших підходів до управління процесом розробки програмного забезпечення. Вона передбачає послідовне виконання кожної фази проекту, починаючи від визначення вимог і завершуючи випуском готового продукту. Основні етапи поділяються на:

– визначення вимог: на цьому етапі встановлюються та документуються всі вимоги до програмного продукту. Важливо чітко зрозуміти потреби замовника та врахувати їх у подальшій розробці;

– проектування: на цьому етапі розробляється детальний план системи, включаючи архітектуру, дизайн інтерфейсу та опис функціональності кожного компонента;

– реалізація: написання коду відповідно до визначених на попередніх етапах вимог та проектування. Цей етап включає в себе створення програмного коду та його тестування;

– тестування: на даному етапі виконується налагодження програмного продукту. Проводяться різні види тестувань, включаючи модульне, інтеграційне та системне тестування для впевненості у коректності роботи всіх компонентів системи;

– впровадження: після успішного тестування програмний продукт готовий до впровадження в реальному середовищі. В цей момент відбувається фактична установка та налаштування системи для кінцевих користувачів;

– супровід та підтримка: після впровадження систему слід постійно підтримувати та вдосконалювати відповідно до потреб користувачів.

Основною перевагою каскадної методології є чітка структура та документація кожного етапу. Вона особливо ефективна в тих випадках, коли вимоги до проекту дуже чітко визначені та малоймовірні зміни в процесі розробки. Однак, цей метод може бути менш ефективним у випадках, коли важко передбачити всі нюанси на початковому етапі, оскільки будь-які зміни потреб вимагають значних зусиль на коригування попередніх етапів. Приклад каскадної моделі продано на рисунку 2.7.

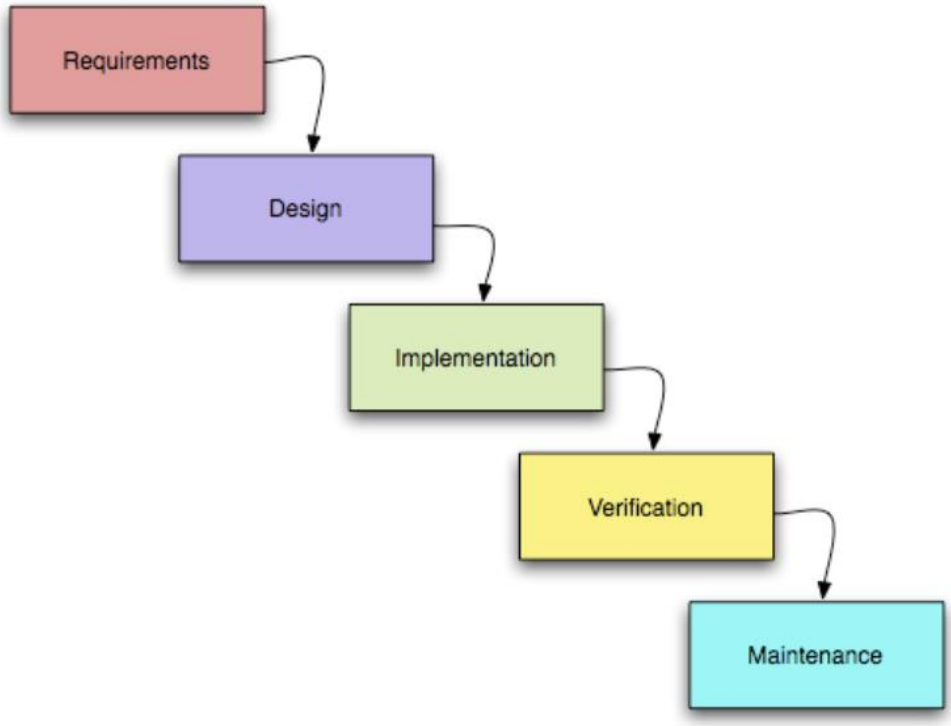

Рисунок 2.7 – Схема каскадної методології розробки

Методологія Agile – це набір підходів та принципів управління процесом розробки програмного забезпечення, які спрямовані на постійну

адаптацію до змін у вимогах та умовах проекту. Вона базується на співпраці команди та гнучкому реагуванні на зміни в процесі розробки. Основні принципи Agile включають:

– співпраця з клієнтом: активна участь замовника у процесі розробки, постійний зворотний зв’язок та зміна пріоритетів відповідно до його потреб;

– ітераційний підхід: розробка ведеться невеликими ітераціями або спрінтами, під час кожного з яких створюється новий функціонал чи виправляються помилки;

– самоорганізація команди: команда має свободу вибору методів та інструментів роботи, а також вирішення труднощів;

– адаптація до змін: Agile підходить до процесу розробки з позиції постійного аналізу та зміни стратегій у відповідь на нові уявлення та вимоги;

– робота з готовим продуктом: На кожному етапі розробки отримується готовий до використання функціонал, що надалі може бути впроваджений.

Методологія Agile підходить для проектів, де важко передбачити всі деталі наперед, або коли вимоги можуть змінюватися під час розробки. Методологія Agile дозволяє ефективно працювати над складними проектами, де потрібна швидка реакція на зміни в умовах або вимогах.

Приклад методології Agile наведено на рисунку 2.8.

Спіральна методологія є гнучкою та ітеративною, комбінуючи принципи каскадної та ітеративної розробки. Основна ідея спіральної методології полягає в тому, щоб постійно оцінювати та аналізувати ризики, пов’язані з проектом, і приймати рішення щодо того, як їх краще уникнути або зменшити. Процес розробки поділяється на кілька ітераційних етапів, кожен з яких включає в себе аналіз ризиків, розробку та тестування.

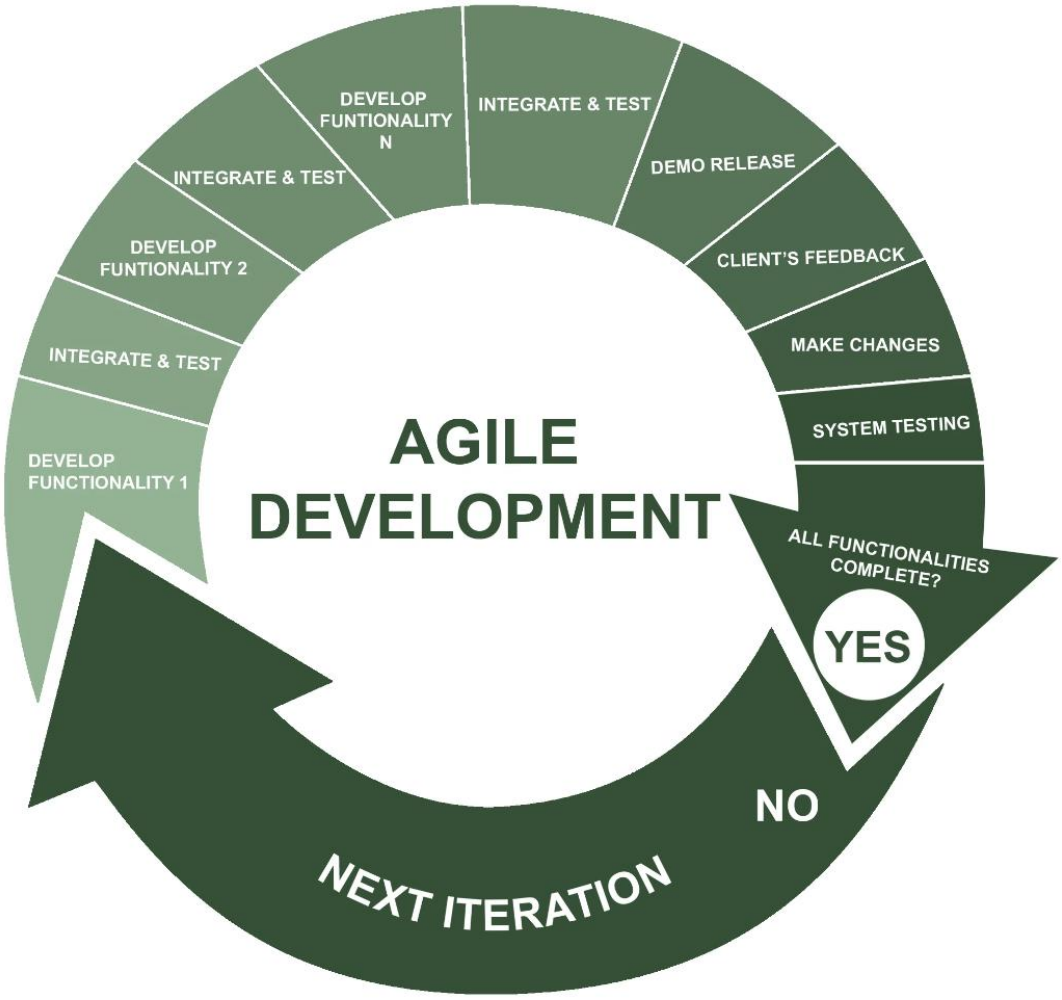

Рисунок 2.8 – Приклад методології Agile

Кожна ітерація «Спіралі» формується як «Обертання» навколо спіралі, пройшовши крок аналізу, розробки та тестування. На кожному кроці можуть бути прийняті стратегічні рішення щодо подальшого розвитку проекту.

Спіральна методологія особливо ефективна для великих та складних проектів, де велика кількість невизначеності та ризиків. Вона дозволяє більш гнучко адаптуватися до змін у вимогах та уникати потенційних проблем у майбутньому приклад спіральної методології наведено на рисунку 2.9. Основні переваги спіральної методології наведено нижче:

– гнучкість: методологія дозволяє адаптуватися до змін у вимогах та ризиках протягом усього проекту;

– аналіз ризиків: постійний аналіз та управління ризиками дозволяє попередити можливі проблеми та уникнути негативних наслідків;

– ітеративність: проект розбивається на невеликі ітерації, що полегшує керування та контроль над процесом розробки;

– зосередженість на якості: кожна ітерація включає тестування, що сприяє підвищенню якості продукту.

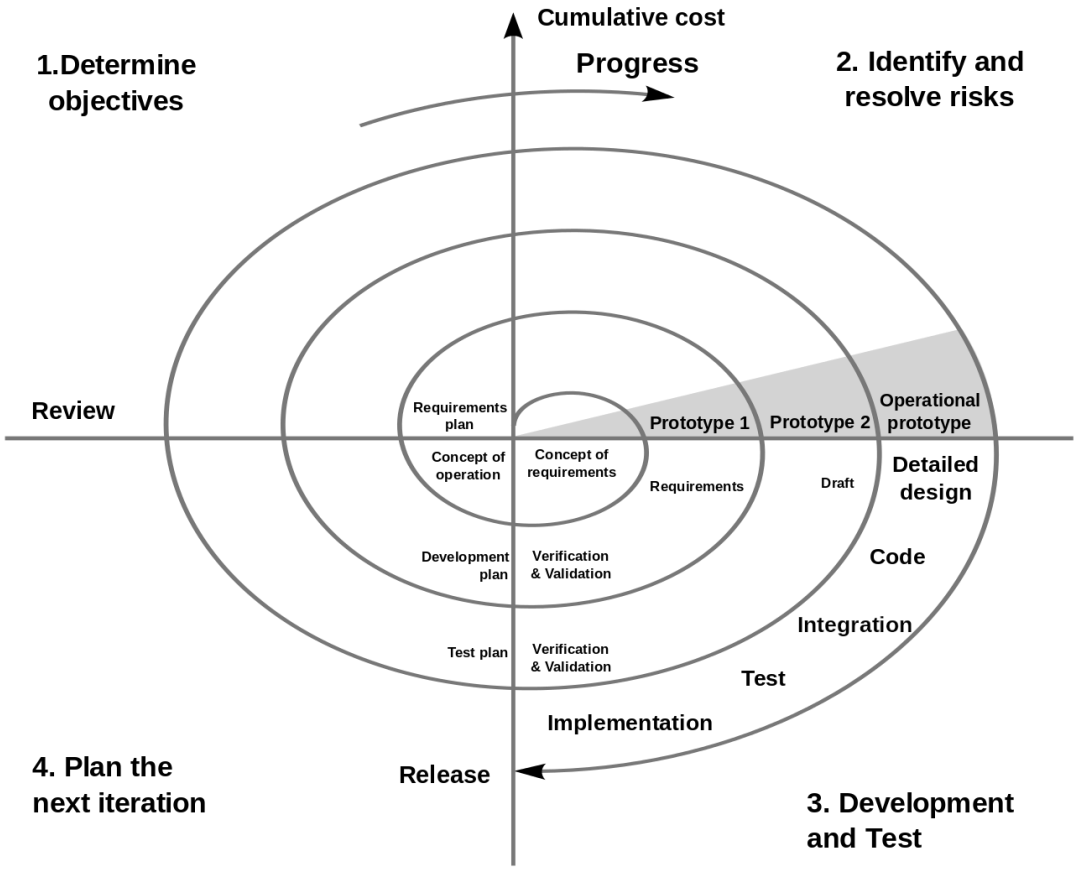

Рисунок 2.9 – Схема спіральної методології

Однак недоліків у спіральної методології також достатньо, вони наведенні нижче:

– складність: реалізація «Спіралі» може бути важливою та вимагати великих зусиль від команди розробників;

– вартість: у зв’язку з постійним аналізом та управлінням ризиками, може виникнути додаткові витрати;

– потреба у досвідчених спеціалістах: реалізація методології вимагає наявності команди з достатнім рівнем досвіду та компетентності;

– не завжди ефективна для невеликих проектів: для менших проектів, де ризики менші, спіральна методологія може бути надмірною та затратної.

3 ВИЯВЛЕННЯ ВИМОГ ТА ПОСТАНОВКА ЗАДАЧІ НА РОЗРОБКУ СИСТЕМИ

3.1 Виявлення вимог

Виявлення вимог на розробку системи визначає ключовий етап у життєвому циклі розробки програмного забезпечення, спрямований на систематичне збирання, аналіз та документування вимог. Ці вимоги визначають потреби та очікування користувачів, мають на меті уточнення та формалізацію завдань, що повинні бути вирішені для задоволення конкретних потреб та досягнення цілей системи [28].

В ході виявлення вимог важливо виділити різні категорії вимог, такі як функціональні, нефункціональні, технічні та бізнес-вимоги. Функціональні вимоги концентруються на конкретних функціях та операціях, які система повинна виконувати. З іншого боку, нефункціональні вимоги охоплюють характеристики, які не є безпосередньо пов’язаними з функціональністю, включаючи аспекти якості та інші аспекти системи.

Технічні вимоги охоплюють аспекти, пов’язані з технічними характеристиками та функціональністю системи. Ці вимоги визначають, як система повинна взаємодіяти з апаратним та програмним забезпеченням, а також які технічні можливості має надавати. Бізнес-вимоги спрямовані на визначення того, як система відповідає потребам та стратегіям бізнесу. Ці вимоги визначаються з урахуванням бізнес-цілей та стратегій компанії.

3.1.1 Функціональні вимоги

Функціональні вимоги у контексті розробки програмного забезпечення представляють собою конкретні функції та операції, які система повинна виконувати для задоволення потреб користувачів та досягнення цілей проекту. Вони описують бажаний функціонал системи та

визначають, як система має взаємодіяти з користувачем, іншими системами чи компонентами.

Під час визначення функціональних вимог важливо враховувати потреби користувачів, бізнес-вимоги та інші чинники, що визначають функціональність системи.

Функціональне моделювання стає важливим інструментом для подальшої розробки системи на основі визначених функціональних вимог. Воно включає в себе створення абстрактних представлень функцій, взаємозв’язків та процесів системи, що допомагає зрозуміти, як система буде функціонувати в реальному середовищі. Функціональні вимоги будуть змодельовані за допомогою методології IDEF-0 (Integrated DEFinition for Function Modeling) [29].

Функціональне моделювання IDEF-0 – методологія, що спрямована на визначення та представлення функціональності системи через графічні моделі. Цей підхід дозволяє зрозуміти структуру та взаємозв’язки між функціями у складних системах.

В IDEF-0 функціональне моделювання базується на використанні функціональних блоків, стрілок та сторін. Функціональні блоки представляють конкретні функції або процеси в системі, стрілки вказують на потік даних чи управління між цими блоками, а сторони дозволяють позначити вхідні або вихідні дані для конкретної функції.

У результаті виконавши моделювання була отримана контекстна діаграма, що подана на рисунку 3.1.

Наступним кроком побудови моделі є декомпозиція контекстної діаграми. Декомпозиція – це процес розбиття складної системи або завдання на менші, більш прості та керовані компоненти чи елементи з метою полегшення розробки, управління та розуміння системи в цілому. Декомпозицію системи об’ємного моделювання подано на рисунку 3.2. Декомпозицію процесу об’ємного моделювання подано на рисунку 3.3.

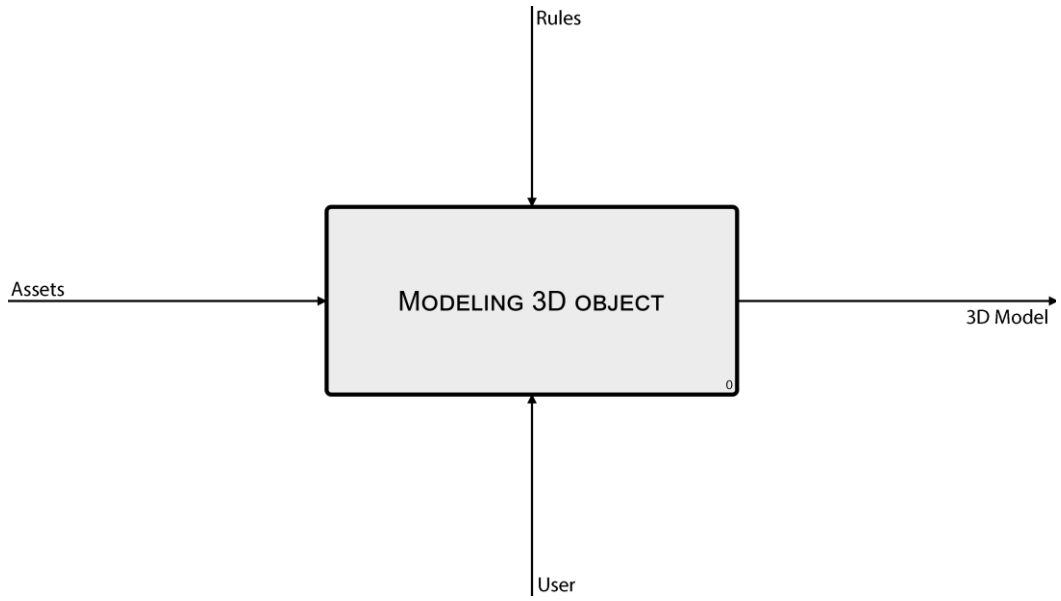

Рисунок 3.1 – Контекстна діаграма у методології IDEF-0

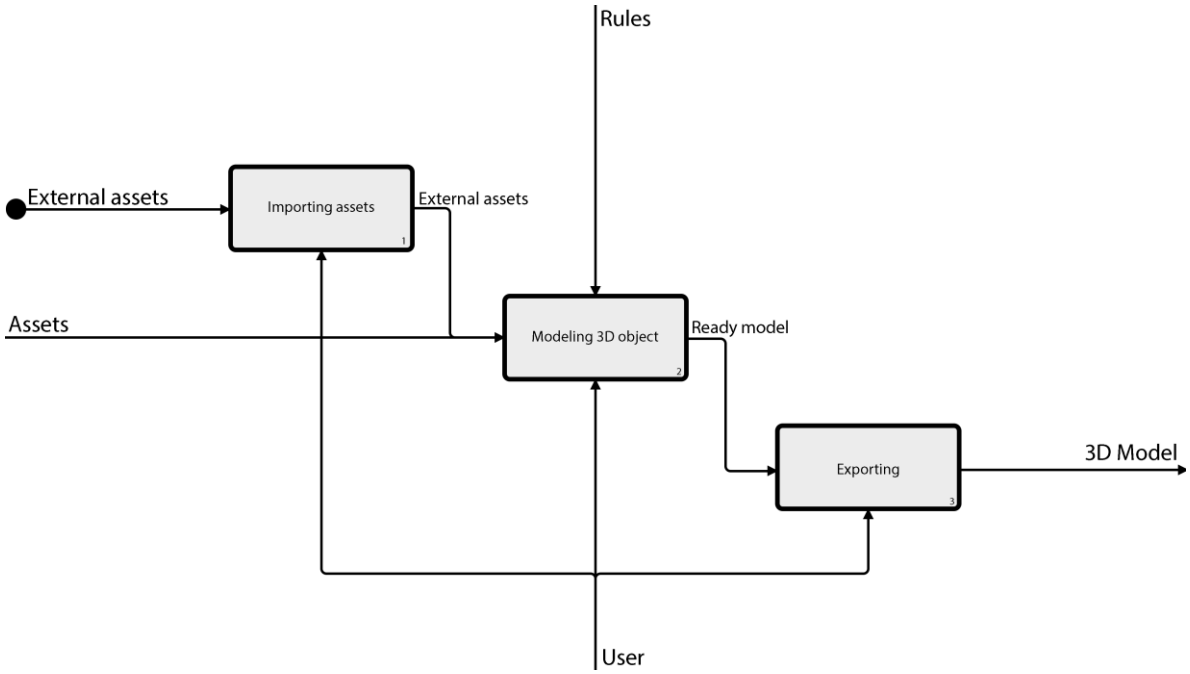

Рисунок 3.2 – Декомпозиція першого рівня діаграми у методології IDEF-0

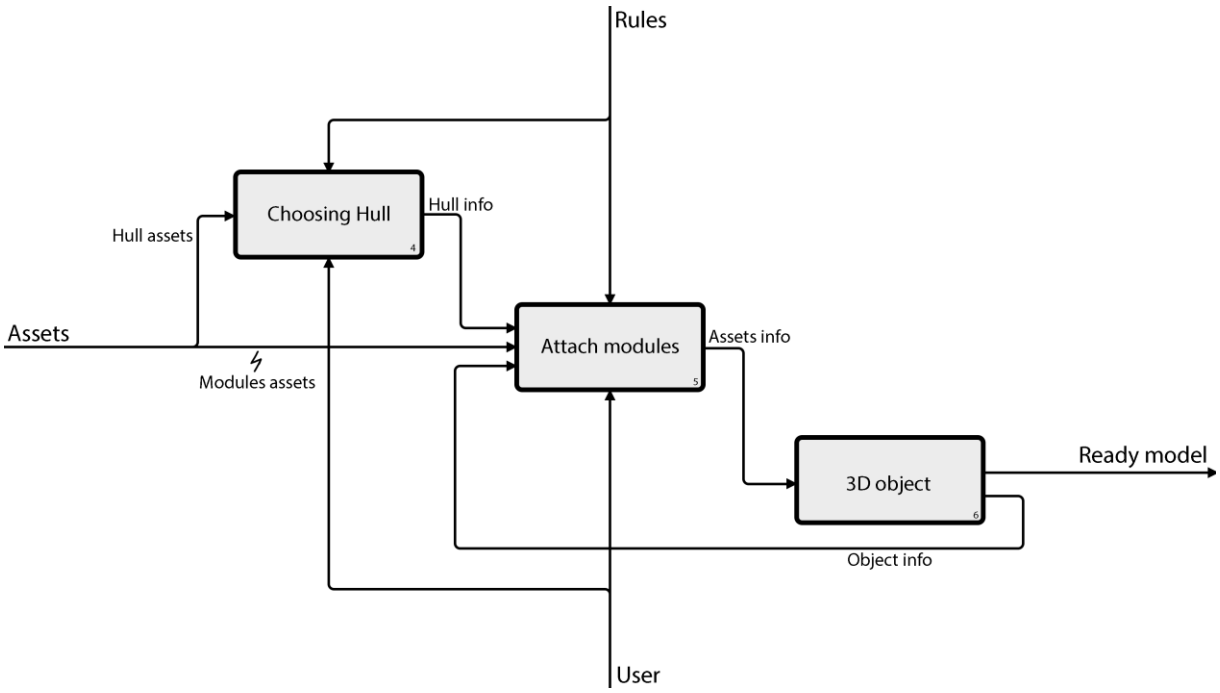

Рисунок 3.3 – Декомпозиція другого рівня діаграми у методології IDEF-0

3.1.2 Нефункціональні вимоги

Нефункціональні вимоги визначають характеристики та обмеження системи, які не стосуються конкретних функцій, але впливають на її якість, продуктивність, безпеку та інші аспекти. Для програми «Система об’ємного моделювання», яка спрямована на моделювання та дизайн, виділяємо такі нефункціональні вимоги: продуктивність; надійність; інтерфейс користувача; сумісність (рис. 3.4).

Продуктивність визначається двома важливими аспектами в контексті системи. По-перше, швидкість роботи передбачає здатність програми здійснювати операції моделювання та редагування тривимірних об’єктів ефективно та без помітних затримок для кінцевого користувача. По-друге, масштабованість визначає, наскільки добре програма впорається з обробкою великої кількості об’єктів та складних сцен, забезпечуючи оптимальну продуктивність при опрацюванні різноманітних елементів та конфігурацій.

Надійність системи також розглядається з точки зору двох ключових аспектів. По-перше, стабільність передбачає, щоб програма була стійкою та спроможною мінімізувати випадки збоїв та ефективно витрачати ресурси. Забезпечення надійності в роботі програми є основоположним для легшого користування системою та попередження можливих негативних впливів на роботу системи. По-друге, надійне збереження даних є невід'ємною частиною функціональності системи, гарантуючи збереження інформації в разі непередбачених ситуацій та забезпечуючи неперервну доступність даних моделей.

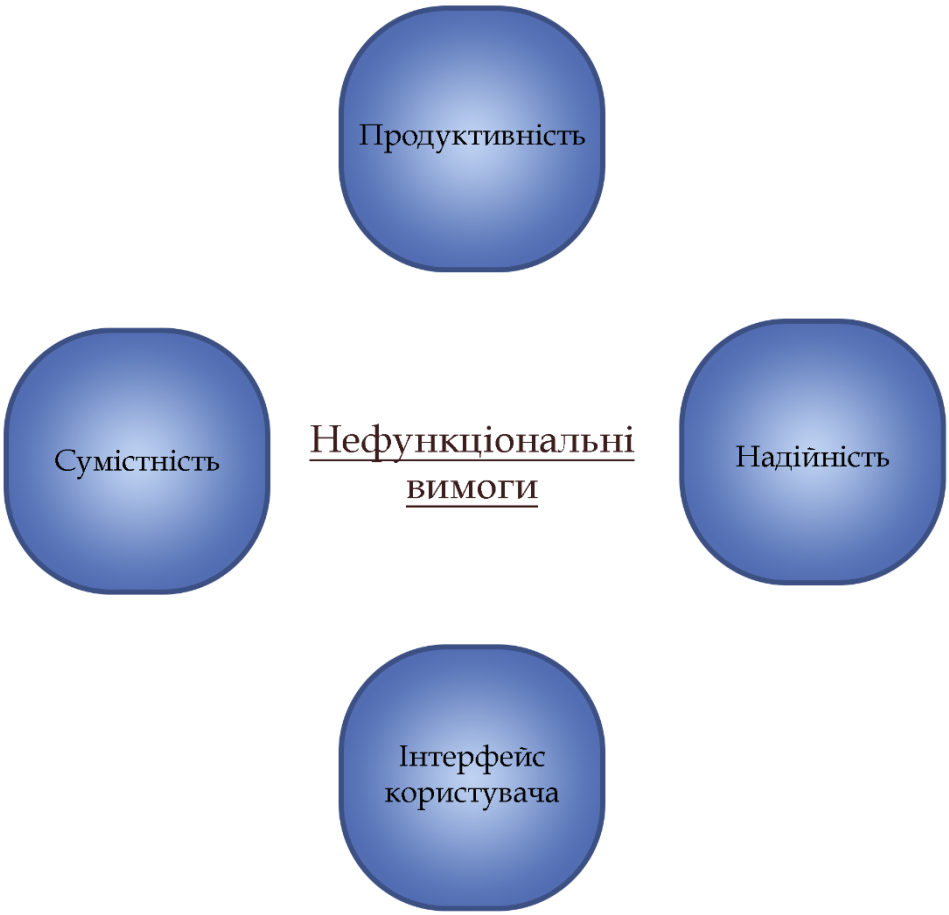

Рисунок 3.4 – Перелік нефункціональних вимог

Інтерфейс користувача має бути інтуїтивним, легким у сприйнятті та зрозумілим для широкого кола користувачів, включаючи тих, у кого немає попереднього досвіду використання аналогічного програмного

забезпечення. Забезпечення інтуїтивного інтерфейсу є ключовим елементом для забезпечення комфортної роботи з програмою та привертання нових користувачів. Наявність розширених можливостей налаштувань дозволяє користувачам адаптувати програму до своїх індивідуальних потреб та вподобань, підвищуючи рівень персоналізації та задоволення від використання програмного забезпечення [30].

Сумісність визначається важливим аспектом - підтримкою різних форматів файлів. Забезпечення можливості імпорту та експорту моделей у різних форматах є критичним для забезпечення сумісності програми з іншими графічними редакторами та інженерними системами. Це важливо з точки зору обміну даними та співпраці між різними платформами та інструментами.

3.1.3 Технічні вимоги

Технічні вимоги для програмного забезпечення включають в себе рекомендації щодо характеристик комп’ютера, на якому система має працювати оптимально. Розглянуті технічні вимоги для ефективного функціонування системи об’ємного моделювання:

Мінімальні вимоги:

– процесор: двоядерний процесор з тактовою частотою не менше 2 ГГц;

– оперативна пам’ять (RAM): мінімум 4 ГБ;

– відеокарта: дискретна відеокарта із підтримкою OpenGL 3.0 або вище;

– місце на жорсткому диску: мінімум 5 ГБ вільного простору.

Рекомендовані вимоги:

– процесор: чотирьохядерний процесор з тактовою частотою 3 ГГц або вище;

– оперативна пам’ять (RAM): рекомендовано 8 ГБ або більше;

– відеокарта: дискретна відеокарта із підтримкою OpenGL 4.0 або вище;

– місце на жорсткому диску: рекомендовано 10 ГБ вільного простору.

Інші вимоги:

– операційна система: підтримка операційних систем Windows 10 або macOS 10.14 і вище;

– монітор: роздільна здатність екрану не менше 1920x1080 пікселів;

– миша та клавіатура: сумісні з стандартами HID (Human Interface Device).

3.1.4 Бізнес вимоги

Основні бізнес-принципи для програмного продукту, який реалізує функціонал моделювання тривимірних об’єктів, націлені на досягнення максимальної безкоштовності та повної відмови від будь-якого фінансового внеску з боку користувачів. Цей стратегічний підхід визначається необхідністю забезпечити вільний доступ та використання програмного продукту, що відповідає принципам некомерційної діяльності та безоплатного надання послуг.

Програма є повністю безкоштовною для всіх користувачів. Немає наміру впроваджувати платні функції або обмеження доступу до будь-яких інструментів. Також програма не містить рекламних модулів чи платних рекламних послуг, що дозволяє користувачам використовувати продукт без будь-яких відволікаючих елементів. Система не взаємодіє з особистими даними користувачів, дотримуючись принципів конфіденційності та не використовує їх для комерційних цілей.

Додатково, важливо відзначити, що розробники програмного продукту абсолютно не зацікавлені у будь-якому отриманні американських гривень чи інших фінансових вигодах. Розробники не будуть активно

залучати або обробляти фінансові кошти від користувачів чи будь-яких комерційних суб’єктів у зв’язку з використанням програмного продукту. Такий підхід підкреслює не ангажованість у фінансових питаннях та визначає пріоритет в наданні безкоштовних та якісних сервісів.

3.2 Вихідна інформація

У рамках системи об’ємного моделювання вихідна інформація визначається як тривимірний об’єкт, що становить основу для подальшого моделювання та творчої роботи. Цей об’єкт буде представлений у форматі FBX або OBJ, що дозволяє зберегти всі необхідні геометричні та текстурні характеристики. Використання цих форматів відкриває можливості для ефективного обміну інформацією та використання створеного об’єкта в різноманітних графічних та технічних середовищах.

Передача вихідної інформації відбудеться одноразово, на завершальному етапі роботи, коли об’єкт буде готовий для використання. Такий підхід сприяє збереженню цілісності та якості моделі, а також уникненню надмірної фрагментації процесу розробки.

Термін видачі інформації буде проводитись одразу, що означає миттєве надання інформації після завершення роботи над тривимірним об’єктом. Це дозволить користувачеві негайно впровадити отриманий об’єкт у своїй творчій діяльності та продовжити роботу без зайвих затримок.

Одержувачем цієї інформації є користувач, який використовує цей тривимірний об’єкт у своїх творчих проектах. Дана система передачі інформації дозволяє забезпечити прозорий та швидкий обмін між системою та користувачем, надаючи можливість користувачам втілювати свої творчі ідеї у тривимірному форматі без зайвих труднощів та затримок.

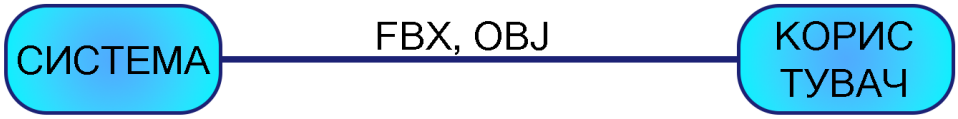

Рисунок 3.5 – Демонстрація вихідних даних

3.3 Вхідна інформація

У рамках системи об’ємного моделювання першою вхідною інформацією є набір «Внутрішні компоненти», який визначає ключові елементи, що складають систему об’ємного моделювання (рис. 3.6).

Ця інформація є основою для подальшої роботи та розвитку проекту. Тип цієї вхідної інформації – «тривимірні об’єкти», оскільки саме їх характеристики, форма та взаємодія будуть визначати результат моделювання.

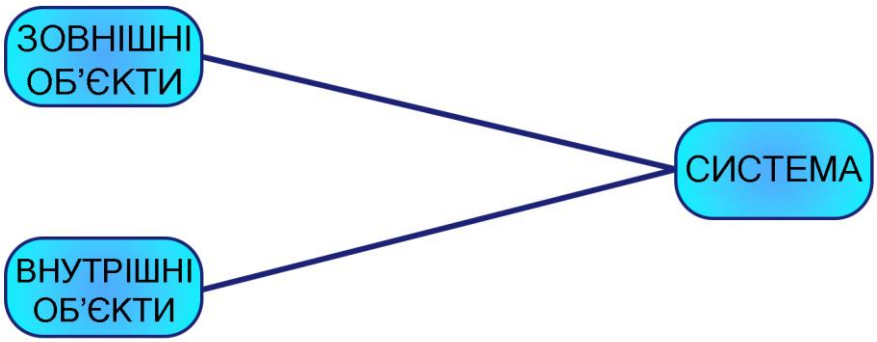

Рисунок 3.6 – Демонстрація вхідних даних

Джерелом цієї вхідної інформації є сама система. Внутрішні компоненти доступні за замовчуванням в межах самої системи об’ємного моделювання і є основою, з якою користувач буде працювати під час творчого процесу. Таким чином, система є початковим джерелом, яке надає вихідні компоненти для подальшого творення тривимірних об’єктів.

Другою вхідною інформацією є «Зовнішні компоненти», що охоплюють елементи, розроблені у зовнішніх середовищах та імпортовані у систему об’ємного моделювання. Такі компоненти можуть бути створені в інших графічних редакторах чи програмах для тривимірного моделювання, а потім інтегровані у робочий процес основної системи.

Тип вхідної інформації – «Тривимірні об’єкти», оскільки зовнішні компоненти також представляють собою тривимірні об’єкти, які будуть використовуватися у процесі подальшого творчого моделювання.

Джерело цієї вхідної інформації може бути будь-яке зовнішнє середовище чи програма, в якій були створені або редаговані ці компоненти. Це може включати, наприклад, інші графічні редактори, спеціалізовані програми для тривимірного моделювання чи навіть бібліотеки готових об’єктів.

4 ПЛАНУВАННЯ ТА ОРГАНІЗАЦІЯ РОЗРОБКИ СИСТЕМИ

Планування та організація розробки системи є важливим етапом у життєвому циклі проекту, що визначає його успішність та ефективність. На цьому етапі визначаються стратегії, методології та інші аспекти, що впливають на подальший процес створення програмного забезпечення.

4.1 Вибір методології розробки

Першочергово, необхідно визначити методологію розробки, яка найкращим чином відповідає специфіці проекту. Методологією розробки системи «Об’ємного Моделювання» обрано каскадну модель розробки як стратегію для реалізації проекту.

Основними критеріями вибору цієї методології стали її простота та адаптивність для індивідуальної роботи.

Оскільки розробників у нас цілий один, каскадна методологія забезпечує послідовність та структурованість у виконанні завдань.

Методологія передбачає розподіл процесу розробки на фіксовані етапи: аналіз, проектування, реалізація, тестування та впровадження.

Такий лінійний підхід дозволяє чітко визначити послідовність робіт та докладно спланувати кожний етап.

Важливо зазначити, що каскадна методологія, хоча й забезпечує структурованість та чіткість процесу, володіє певними обмеженнями. Зокрема, це стосується гнучкості внесення змін під час розробки.

Через це обмеження, необхідно передбачити та проаналізувати всі вимоги ще на етапі планування проекту, з метою уникнення можливих труднощів під час проектування системи.

4.2 Вибір програмного забезпечення

В процесі розробки передбачено використання рушію Unity 3D як основного інструменту для втілення поставлених завдань. Обрання готового рушію, такого як Unity 3D, визначено як пріоритетна альтернатива у зв’язку із стрімкими термінами реалізації проекту та прагненням економії часу, яка може бути витрачена на створення власного рушію з нуля.

Unity 3D володіє низьким порогом входження та надає можливість зосередитися безпосередньо на проектуванні системи, уникаючи зайвих затримок, пов’язаних із налаштуванням власного рушію. Його інтуїтивно зрозумілий та простий інтерфейс, складений з різноманітних вікон, які відображають різні аспекти розробки, забезпечує зручність та ефективність роботи.

Особливо важливим є те, що Unity 3D дозволяє взаємодіяти одночасно з різними елементами середовища, сприяючи швидкому та ефективному прогресу у роботі над проектом. Це підвищує продуктивність та сприяє зручності в розробці, роблячи Unity 3D оптимальним вибором для реалізації системи «Об’ємного Моделювання».

4.3 Стратегія розробки компонентів системи та їх функціональний тип

В даному розділі визначається концептуальний образ елементів програми у порівнянні з існуючими аналогами. Пріоритетним завданням є створення програми, що вирізняється простотою сприйняття та легкістю в експлуатації. В основу цього підходу покладено спостереження за характеристиками «Внутрішніх ігрових конструкторів», визнаних як оптимальний аналог з погляду інтуїтивної структури.

4.3.1 Узагальнений вигляд

Першочерговими критеріями розробки є зрозумілість та легкість використання програми. Основною вимогою до програми є її зрозумілість та легкість в експлуатації, а ці аспекти ефективно втілені у «Внутрішніх ігрових конструкторах», які вирізняються інтуїтивно зрозумілою структурою, що відкриває шлях до використання аналогічної інтуїтивної структури у розробці проекту.

У подальшому планується взяти за основу зовнішній вигляд ігрових конструкторів для додаткового аналізу та вдосконалення концептуального вигляду елементів програми. Цей підхід дозволяє врахувати високий рівень естетичності та зручності у взаємодії з користувачем, роблячи програму не лише ефективним інструментом, але й досвідом, що приносить задоволення від використання.

4.3.2 Вигляд інтерфейсу користувача

По-друге, простота інтерфейсу орієнтована на акцентування уваги користувача на основних елементах управління та моделі. Деталізація цих характеристик виявляється ефективною при аналізі систем тривимірного моделювання, ігрових конструкторів та систем моделювання на основі штучного інтелекту.

Системи штучного інтелекту виходять за рамки потреб системи, тому їх вивчення та застосування елементів відхиляється. Замість цього, аналізуючи системи тривимірного моделювання та ігрові конструктори, визначено переваги простоти їхнього інтерфейсу та методів роботи.

Детальніше, інтерфейс систем тривимірного моделювання визначено як джерело для структури інтерфейсу, оскільки такі системи можуть забезпечити інформативність без перевантаження елементів інтерфейсу. Системи 3D-моделювання обрано як основу для визначення структури

інтерфейсу з урахуванням вимог до інформаційної наповненості та невисокої складності використання.

4.3.3 Алгоритми моделювання

Алгоритми моделювання – важливий аспект розробки, які, хоча і знаходяться поза областю прямого спостереження користувача, мають визначальний вплив на функціональність програми. Розглядаючи можливі альтернативи, можна відзначити ігрові конструктори, але вони виявляються недостатньо функціональними для поставлених задач.

У цьому контексті розглядається можливість використання алгоритмів з систем тривимірного моделювання та інженерних систем. Інженерні системи відрізняються великим функціоналом, але їх реалізація виявляється складною. З іншого боку, функціонал тривимірних систем моделювання є менш складним, що забезпечує шлях до конструктивного об’єднання елементів обох систем.

Таким чином, буде побудовано алгоритми моделювання, використовуючи синергію між системами тривимірного моделювання та інженерними системами, надаючи вагу їхньому функціоналу та забезпечуючи оптимальний баланс між складністю та ефективністю реалізації.

4.4 Моделювання Діаграми Потоків Даних: Структура та Взаємодія Компонентів Системи

Моделювання Діаграм Потоків Даних буде здійснюватися відповідно до стандартів DFD, забезпечуючи відповідність та структурованість у візуалізації потоків і обробки даних в системі [31].

DFD (Діаграми Потоків Даних) є графічним методом для моделювання систем, який використовується для візуалізації потоків і обробки даних у системі. Основними компонентами DFD є:

– процеси: представляють дії або операції, які виконуються над даними;

– потоки даних: визначають переміщення даних в системі і представлені стрілками. Потоки даних відображають напрямок передачі даних від одного елемента до іншого;

– сховища даних: вказують на місце, де зберігаються або використовуються дані;

– зовнішні сутності: вказують на елементи або системи, які взаємодіють з системою, але не є частиною неї.

DFD використовується для аналізу та проектування систем, допомагаючи розкрити основні елементи та їх взаємодію. Ця модель дозволяє розуміти, як дані обробляються та передаються в системі, і є важливою частиною процесу розробки програмного забезпечення. Діаграма верхнього рівня наведена на рисунку 4.1.

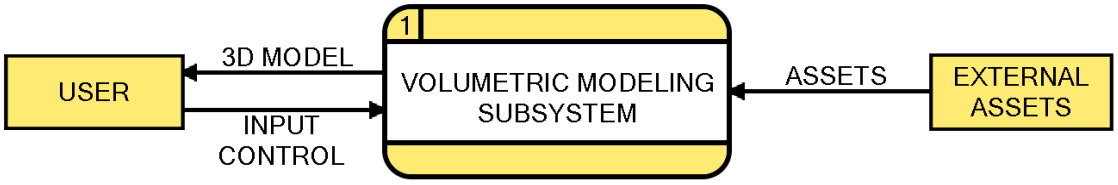

Рисунок 4.1 – Діаграма потоків даних, верхній рівень

Наступним кроком побудови моделі є декомпозиція діаграми потоків даних. Декомпозиція – це процес розбиття складної системи або завдання на менші, більш прості та керовані компоненти чи елементи з метою полегшення розробки, управління та розуміння системи в цілому.

Декомпозицію системи об’ємного моделювання подано на рисунку 4.2.

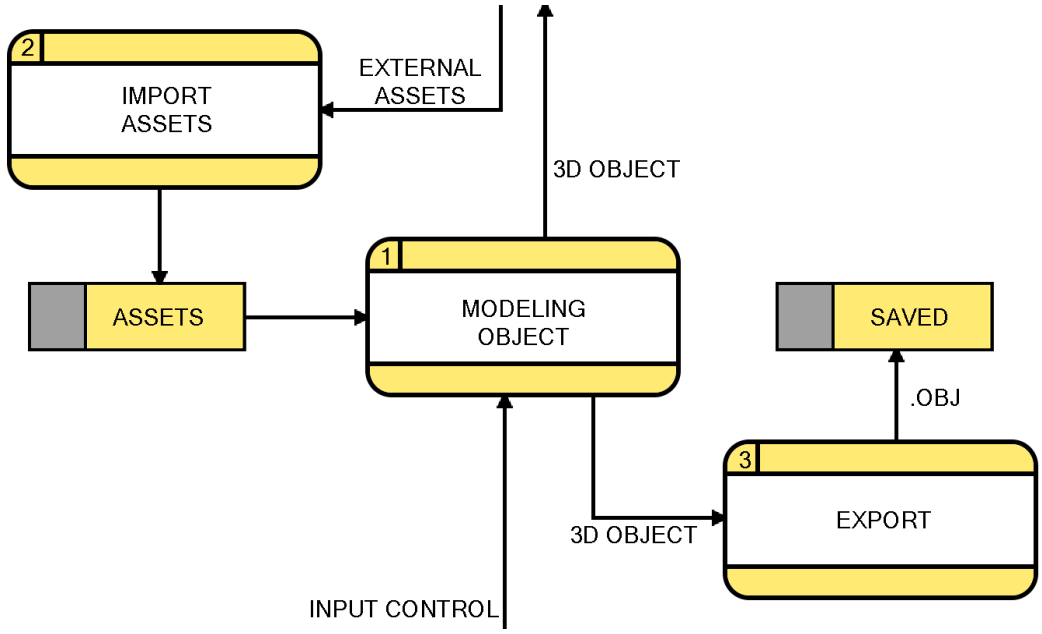

Рисунок 4.2 – Декомпозиція системи об’ємного моделювання

4.5 UML Діаграми: Засоби Визначення, Взаємодія та Архітектура Системи

UML (Unified Modelling Language) є стандартом для визначення, візуалізації, специфікації та документування архітектури програмних систем. UML дозволяє команді розробників, аналітикам та іншим учасникам проекту використовувати єдиний мовний засіб для спілкування та розуміння всіх аспектів системи. Це сприяє полегшенню процесів аналізу, проектування та розробки програмного забезпечення. Основні складові UML включають [32]:

- Use Case Diagram;
- Class Diagram;
- Sequence Diagram;
- Activity Diagram;
- UML state machine.

4.5.1 Побудова Sequence diagram

Діаграма Послідовності у рамках UML є графічним засобом моделювання взаємодій між об’єктами в системі в часовому порядку. Вона надає зображення послідовності подій, які виникають під час виконання конкретної функціональності, відображаючи об’єкти, які взаємодіють між собою. Основні елементи Sequence Diagram:

Sequence Diagram прецеденту «Створення моделі» розроблюваної системи подано на рисунку 4.3.

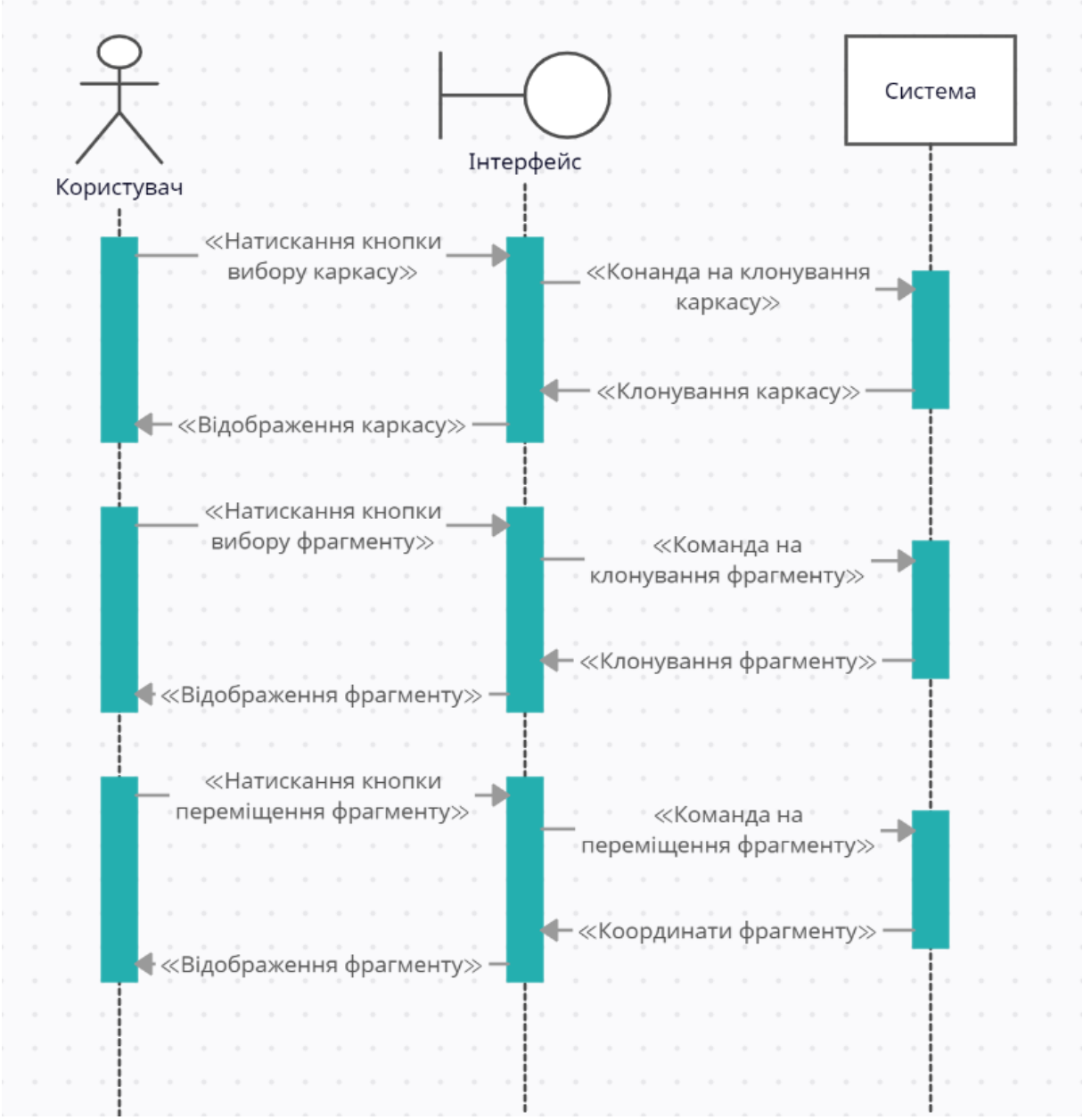

Рисунок 4.3 – Sequence Diagram прецеденту «Створення моделі»

Об’єкти в контексті діаграми послідовності представляють учасників системи, які взаємодіють між собою.

Ці об’єкти розміщені вертикально та є візуальним зображенням суб’єктів у системі.

Діаграма послідовностей відображає вертикальними лініями життєвий цикл, на якому зображені процеси або об’єкти.

Послідовності подій визначають конкретні дії та взаємодії між об’єктами. Стрілки, які з’єднують об’єкти, вказують напрямок взаємодії, а підписані подіями, вони відображають порядок виникнення подій.

4.5.2 Побудова Activity Diagram

Діаграма діяльності є графічним інструментом моделювання, який використовується для представлення послідовності дій або діяльності, що відбуваються в системі або процесі.

Ця діаграма дозволяє відобразити, як об'єкти співпрацюють між собою в рамках конкретної діяльності чи послідовності дій.

У діаграмі діяльності сполученими стрілками налічують такі основні елементи:

– закругленими прямокутниками позначаються дії, вузол управління – координує потоки дій;

– ромбами позначають рішення, вузол визначає правила розгалуження, у вузол входить лише один зв’язок, а виходить два або більше;

– риски позначають початок розподілу або кінець об’єднання паралельних діяльності.

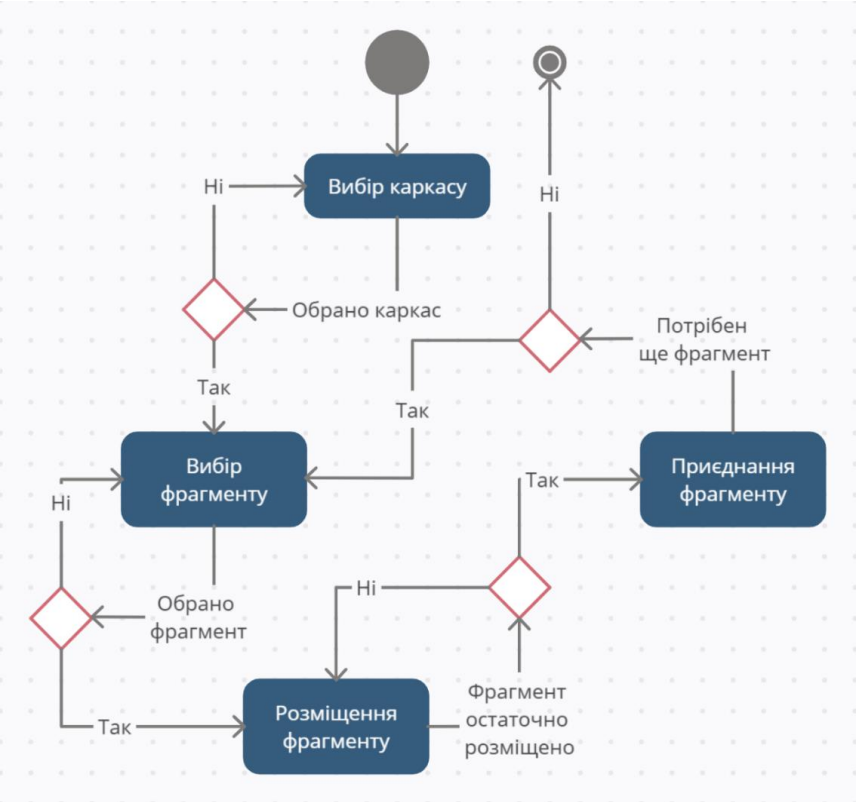

Рисунок 4.4 – Activity Diagram

4.5.3 Побудова UML state machine

UML State Machine, або діаграма станів, є інструментом моделювання, який визначає динаміку системи, подаючи зміни станів об’єкта відносно подій та умов. У програмуванні, де керування подіями є ключовим, концепція скінченного автомата виявляється важливою, спрямовуючи обробку подій залежно від їх типу або поточного стану системи. Це дозволяє зменшити кількість можливих варіантів виконання коду та спрощує умови використання умовного розгалуження.

Діаграма станів у UML представляє собою орієнтований граф, де вузли визначають стани системи або програми, а зв’язки вказують переходи між цими станами. Стани позначаються закругленими прямокутниками зі змістовною назвою. Зв’язки визначають напрямок переходу між станами, відображаючи, який стан може слідувати за іншим, або представляючи множину можливих станів. У кожній діаграмі важливо визначити вхідний

стан (зафарбований круг без назви) та вихідний стан (зафарбований круг у крузі).

Діаграму станів прецеденту «Створення моделі» наведено на рисунку 4.5.

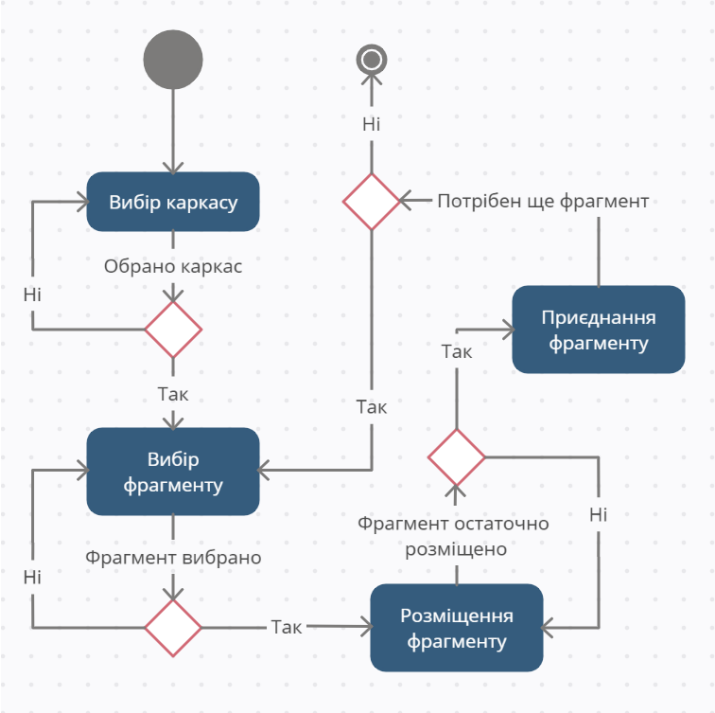

Рисунок 4.5 – Діаграма станів прецеденту «Створення моделі»

5 РЕАЛІЗАЦІЯ СИСТЕМИ ОБ’ЄМНОГО МОДЕЛЮВАННЯ

Реалізація системи – це критичний етап в життєвому циклі розробки, де концепції та ідеї перетворюються в конкретний функціональний продукт. Реалізація системи вимагає значних зусиль та ресурсів.

На етапі розробки системи особливо важливо вивчити відповідність концепції системи з поставленими метами проекту. Важливо переконатися, що розробка відповідає конкретним завданням та вимогам проекту.

Отже резюмуючи, має бути розроблена система за методологією каскадної моделі на рушію Unity 3D.

Система буде включати елементи, взяті з узагальненого вигляду «Внутрішні Ігрові Конструктори», матиме інтерфейс користувача, орієнтований на «Системи Тривимірного Моделювання», і алгоритми моделювання, що поєднують у собі елементи «Системи Тривимірного Моделювання» та «Інженерні Системи».

Також етап реалізації має враховувати можливості майбутнього розвитку системи.

Гнучкість та масштабованість системи грають важливу роль у забезпеченні довгострокового успіху та адаптації до змінних умов.

5.1 Проектування Інтерфейсу Користувача

Необхідно розробити інтерфейс, який буде інтуїтивно зрозумілим та легким у сприйнятті для широкого спектру користувачів, включаючи тих, які не мають попереднього досвіду використання аналогічного програмного забезпечення.

Основною метою є забезпечення можливості користувачам працювати з програмою без потреби в додатковому навчанні.

5.1.1 Візуальне проектування прототипу інтерфейсу

Створення прототипу інтерфейсу є важливим етапом в розробці програмного продукту, оскільки дозволяє заздалегідь визначити та перевірити ключові аспекти його дизайну та функціональності. Прототип надає можливість відобразити концепції ідеального інтерфейсу та здійснити взаємодію з елементами програми на ранніх етапах розробки. Цей процес сприяє виявленню та виправленню можливих проблем чи неузгоджень ще до завершення фінальної реалізації, що економить час та ресурси. Прототип Інтерфейсу наведено на рисунку 5.1.

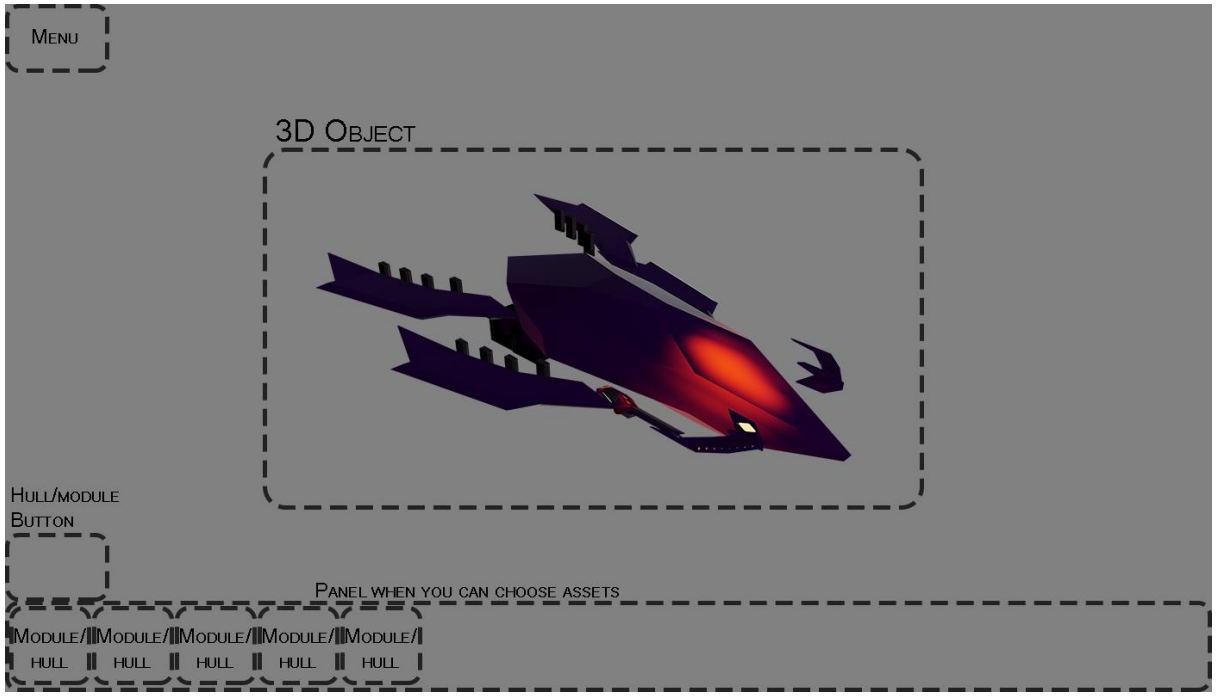

Рисунок 5.1 – Прототип інтерфейсу

Хоча інтерфейс, створений на етапі прототипування, може не завжди визначати остаточний та оптимальний вигляд програми, він важливий для того, щоб задовольнити поточні потреби та вимоги проекту. Інтерфейс може бути об’єктом ітерацій та вдосконалення на подальших етапах розробки. Це дозволяє задовольнити поточні потреби та забезпечити можливість

взаємодії з користувачем, одночасно залишаючись гнучким для майбутніх змін та вдосконалень.

5.1.2 Розробка Інтерфейсу користувача

Інтерфейс користувача розроблено на основі рушія Unity 3D як основна платформа для розробки. Цей вибір обумовлений його здатністю повністю відповідати специфікаціям та вимогам проекту. Unity забезпечує надійну та ефективну основу для реалізації інтерфейсу, а його функціональні можливості сприяють створенню зручного та інтуїтивно зрозумілого інтерфейсу для користувачів. Матеріали наведено нижче.

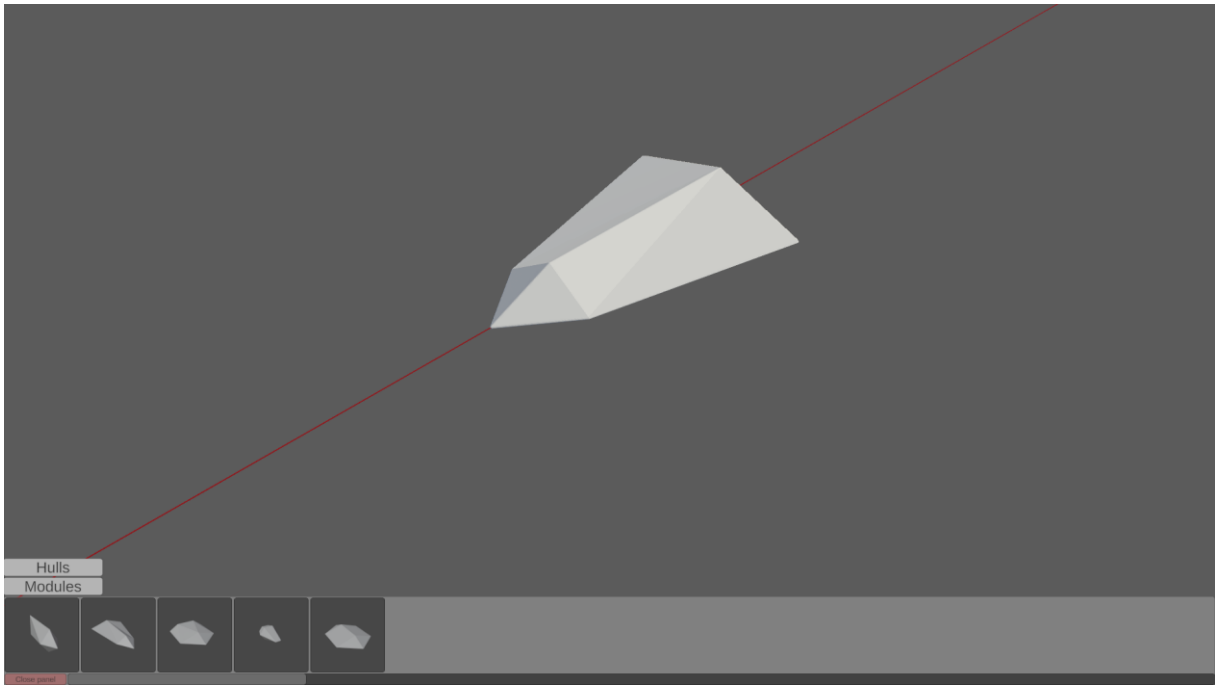

Рисунок 5.2 – Розроблений інтерфейс користувача

Лістинг 5.1 – Програмний код елементу Інтерфейсу

```
public void OpenHullPanel()
{
    if (hullPanel.activeSelf == false)
    {
        modulePanel.SetActive(false);
        hullPanel.SetActive(true);
    }
```

```
    else
    {
        hullPanel.SetActive(false);
    }

}
```

5.2 Проектування основної механіки системи

Процедура створення тривимірного об’єкта включає в себе етап підбор каркасу з панелі, що налічує доступні каркаси, і наступний етап прикріплення до нього додаткових фрагментів. Програмну реалізацію наведено у лістингу 5.2.

Лістинг 5.2 – Програмна реалізація вибору компонентів

```
public void HullInstantiate()
{
    try
    {
        //TODO - print message "U sure want to Delete?"

        Destroy(GameObject.FindWithTag("cloned hull"));
    }
    catch (Exception e)
    {
        Debug.LogError(e);
    }

    hullClone = Instantiate(hullObj.Hull, Vector3.zero,
Quaternion.Euler(Vector3.zero));
    hullClone.tag = "cloned hull";
    hullClone.transform.Find("default").AddComponent<MeshCollider>();
}

public void OnClickDown()
{
    instance.SetSelectedModule(Instantiate(moduleObj.Module));

instance.GetSelectedModuleForce().transform.Find("default").AddComponent<Me
shCollider>().enabled = false;
    instance.GetSelectedModuleForce().AddComponent<Outline>();
    instance.GetSelectedModuleForce().GetComponent<Outline>().OutlineColor
= Color.red;
}

private void ModulePlacer()
{
    Ray ray = camMain.ScreenPointToRay(Input.mousePosition);

    if (Input.GetMouseButton(0))
    {
        if (Physics.Raycast(ray, out var hit, Mathf.Infinity) &&
instance.GetSelectedModule() != null)
```

```
        {
            instance.GetSelectedModule().transform.position = hit.point;
        }
    }

    if (Input.GetMouseButtonUp(0))
    {
        if (Physics.Raycast(ray, out var hit, Mathf.Infinity))
        {
            if (!hit.transform.parent.CompareTag("cloned hull"))
                Destroy(instance.GetSelectedModule());
            if (hit.transform.parent.CompareTag("cloned hull"))

instance.GetSelectedModule().transform.SetParent(hit.transform);
        }

    }
}
```

Додатково розроблено модуль, що дозволяє вибирати вже прикріплені модулі та переміщувати їх, додатково він підсвічує вибраний модуль. Демонстрація вибраного модуля наведена на рисунку 5.3.

Програмну реалізацію наведено у лістингу 5.3.

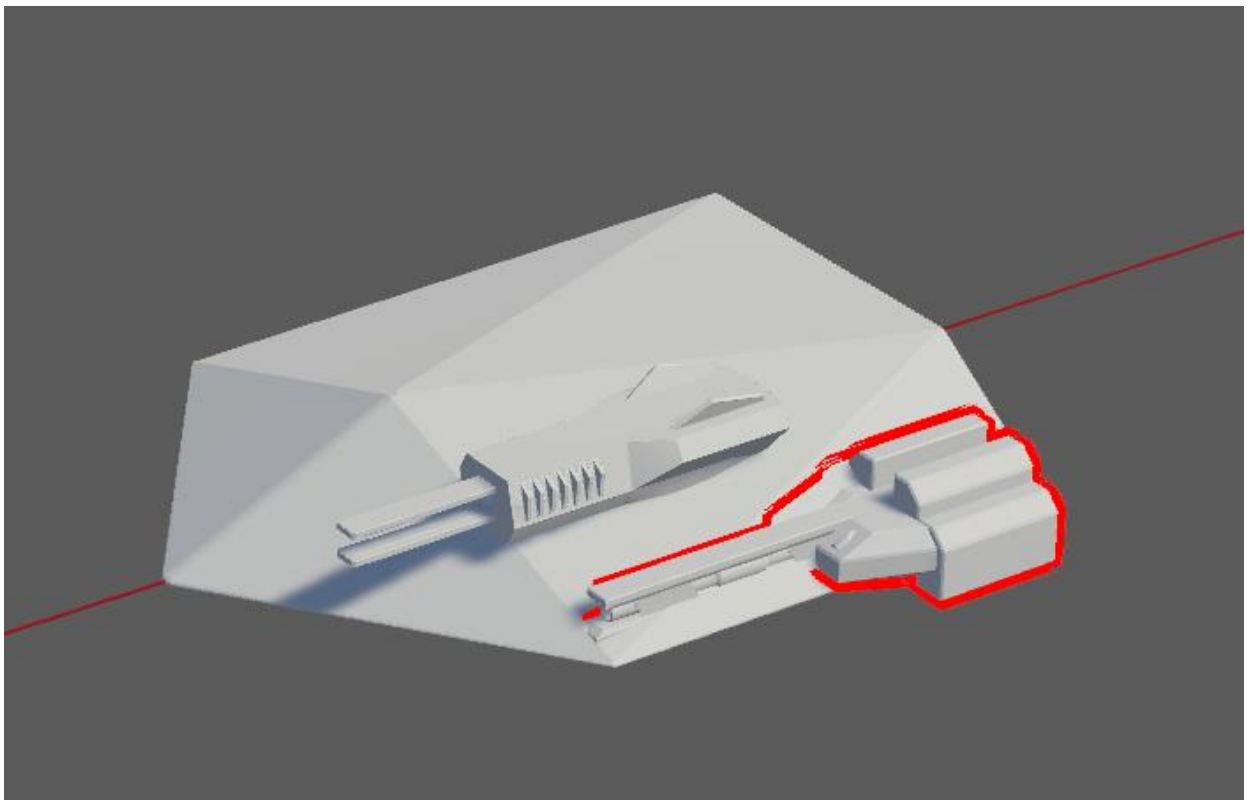

Рисунок 5.3 – Вибраний модуль підсвічено

Лістинг 5.3 – Програмна реалізація вибору модуля

```
private void Update()
{
    Ray ray = Camera.main.ScreenPointToRay(Input.mousePosition);

    if (Input.GetMouseButtonDown(0) && Physics.Raycast(ray, out var hit) &&
        !hit.transform.parent.CompareTag("cloned hull") &&
!hit.transform.CompareTag("MainCamera"))
    {
        SetSelectedModule(hit.transform.parent.gameObject);
    }
}

public void SetSelectedModule(GameObject obj)
{
    try
    {
        currentModule.GetComponent<Outline>().enabled = false;

currentModule.transform.Find("default").GetComponent<MeshCollider>().enable
d = true;
    }
    catch
    {
        // ignored
    }

    currentModule = obj;

    try
    {
        currentModule.GetComponent<Outline>().enabled = true;

currentModule.transform.Find("default").GetComponent<MeshCollider>().enable
d = false;
    }
    catch
    {
        // ignored
    }
}
```

5.3 Перспективні напрямки розробки: стратегії та майбутні розширення

У подальшому розвитку системи передбачається розширення базових модулів, що доступні користувачу одразу. Серед ключових аспектів подальшого розвитку знаходяться розширення багатофункціональної бібліотеки об’єктів та інструментів, що суттєво сприятиме удосконаленню процесу творчості. Далі, планується вдосконалення засобів, які дозволять накладати текстурні елементи на тривимірні моделі, що надасть

користувачам можливість збагачення візуального вигляду їх арт-об’єктів реалістичними та деталізованими текстурами.

Також у планах входить розробка інструментів для зміни освітлення сцени, надаючи користувачам додаткові можливості впливу на атмосферу та віртуальних оточень. Передбачається впровадження роботи з шейдерами, що дозволить більш детально керувати візуальними аспектами об’єктів та ефектами в сцені.

За для підтримки взаємодії та обміну творчими ресурсами, планується впровадження маркетплейсу у мережі. Цей функціонал дозволить користувачам публікувати свої арт-об’єкти, а також здійснювати обмін та взаємодію з іншими учасниками спільноти, сприяючи активному обміну ідеями та ресурсами у віртуальному середовищі.

Додатково буде проведено оптимізацію – процес оптимізації включатиме аналіз та вдосконалення алгоритмів обчислень, оптимізацію роботи графічного двигуна, а також усунення можливих джерел надмірного навантаження. Оптимізація має на меті забезпечити користувачам швидший та безперебійний досвід взаємодії з системою, підвищуючи загальну продуктивність та забезпечуючи оптимальне використання ресурсів обчислювальної та графічної систем.

Також буде створено модуль допомоги та підтримки, в цілому, цей аспект орієнтується на надання ефективної підтримки користувачам системи, зокрема новачкам, для полегшення їхнього введення в роботу з продуктом. Розглядається впровадження системи підтримки, що забезпечить користувачів оперативною інформацією та рішенням типових проблем. Окрім цього, планується створення детальної документації, яка охопить всі аспекти використання системи, та розробка відео-уроків, які наддадуть користувачам візуальні та інтерактивні пояснення. Це сприятиме швидшому освоєнню функціоналу та підвищить загальний рівень зручності та задоволення від використання системи.

ВИСНОВКИ

У ході розробки системи було визначено і враховано ключові аспекти, спрямовані на створення інноваційного та функціонального продукту. Оглянуто задачі моделювання та проектування, об’єкту моделювання для інформаційних технологій проектування. Описано технології проектування та оглянуто системи-аналоги моделювання.

Розглянуто види систем моделювання та складено таблицю критичних показників. Розглянуто системи програмування та описано недоліки та переваги. Описано види методологій, їх різновид та доцільність використання.

Виявлено вимоги до системи об’ємного моделювання, поставлено функціональні, нефункціональні технічні та бізнес вимоги. Описано вхідну та вихідну інформацію системи.

Обрано методологію розробки – це каскадна модель. Основними критеріями вибору цієї методології стали її простота та адаптивність для індивідуальної роботи. Програмним забезпеченням обрано – Unity 3D. Unity 3D володіє низьким порогом входження та надає можливість зосередитися безпосередньо на проектуванні системи, уникаючи зайвих затримок, пов’язаних із налаштуванням власного рушію. Описано стратегії розробки компонентів системи, а саме розглянуто узагальнений тип об’єктів, вигляд інтерфейсу користувача та алгоритми моделювання.

Побудовано Діаграми Потоків Даних, UML Діаграми, а саме Sequence diagram, Activity diagram, UML state machine.

Реалізовано систему об’ємного моделювання, спроектовано та розроблено інтерфейс користувача, розроблено основну механіку роботи системи об’ємного моделювання та визначено перспективні напрямки розробки.

За результатами дослідження опубліковано тези доповіді в матеріалах міжнародної науково-практичної конференції «Distance learning: problems,

ways of development and the latest technologies» (December 25-27 2023, Munich, Germany) [4].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Системи 3D-моделювання / Пальчевський Б. О., Валецький Б. П., Вараніцький Т. Л. Луцьк, 2016. 176 с.

2. Mastering 3D-Modeling: A Practical Guide for Artists and Designers / Talmage McLaurin, 2020. 534 p.

3. Numerical Sculpting: Volumetric Modelling Tools for In Place Spatial Additive Manufacturing / I. Mitropoulou, I. Ariza, M. Bernhard [etc.]. In: Gengnagel C., Baverel O., Burry J. [etc.] (eds) Impact: Design With All Senses. DMSB 2019. Springer, Cham. URL: https://doi.org/10.1007/978-3-030-29829-6_11 (дата звернення 17.12.2023).

4. Задоров В.Б. Підхід до створення технології попереднього системного проектування комп'ютерних інформаційних систем підприємств // Інформаційні технології проектування, 2010. №1. С. 56–63. URL: http://mdcs.knuba.edu.ua/article/view/48966 (дата звернення 25.12.2023).

5. Бутирін М. С., Безкоровайний В. В. Оцінка варіантів проєктних рішень при створенні системи об'ємного моделювання // The XV International Scientific and Practical Conference "Distance learning: problems, ways of development and the latest technologies", December 25-27 2023, Munich, Germany. P. 299–305. URL: https://eu-conf.com/events/distance-learning-problems-ways-of-development-and-the-latest-technologies/ (дата звернення 27.12.2023).

6. Beskorovainyi V. Combined method of ranking options in project decision support systems // Innovative Technologies and Scientific Solutions for Industries. 2020. No 4 (14). P. 13–20. URL: http://journals.uran.ua/itssi/article/view/ITSSI.2020.14.013 (дата звернення 17.12.2023).

7. Beskorovainyi V. V., Petryshyn L. B., Shevchenko O. Yu. Specific subset effective option in technology design decisions // Applied Aspects of Information Technology. 2020. Vol. 3. No. 1. P. 443–455. URL:

https://aait.op.edu.ua/?fetch=articles&with=info&id=40 (дата звернення: 16.04.2020).

8. Beskorovainyi V., Berezovskyi H. Identification of preferences in decision support systems // ECONTECHMOD. 2017. Vol. 06. №4. P. 15–20. URL: https://openarchive.nure.ua/items/9a27c447-47cc-44b0-b0c0-e6fffaaccb10

9. Beskorovainyi V., Berezovskyi H. Estimating the properties of technological systems based on fuzzy sets // Innovative technologies and scientific solutions for industries. 2017. № 1 (1). С. 14–20. DOI: https://doi.org/10.30837/2522-9818.2017.1.014

10. Beskorovainyi V., Kolesnyk L., Russkin V. Decision making support under conditions of incomplete consistency of expert advantages // Innovative integrated computer systems in strategic project management": Collective monograph edited by I. Linde. European University Press. Riga: ISMA, 2022. P. 16-26. URL: https://mmp-conf.org/documents/archive/monography2022.pdf (дата звернення 18.11.2022 р.).

11. Лотошинська Н. Д., Ізонін І. В. Технології 3D-моделювання в програмному середовищі 3ds Max з дисципліни «3D-Графіка». Львів: Львівська політехніка, 2020. 216 с.

12. Чехлов Д. V-Ray для Autodesk Maya. Посібник з візуалізації. ДМК Прес. 2020. 808 с.

13. Лоттер Р. Blender: новий рівень майстерності. ДМК Прес. 2022. 452 с.

14. Герасименко О. І. Моделювання в AutoCAD 2021 Двовимірні та тривимірні побудови. ДМК Прес. 706 с. URL: https://grenka.ua/409687/ modelirovanie-v-autocad-2021-dvumernye-i-trekhmernye-postroeniya (дата звернення 17.12.23).

15. Офіційний сайт Artbreede. URL: https://www.artbreeder.com (дата звернення 17.12.23).

16. Стріміть як художник: 10 способів створити свій власний шлях / Остін Клеон, 2019. 203 с.

17. Цифрова платформа: інформаційні технології в соціокультурній сфері. URL: http://infotech-soccult.knukim.edu.ua/ (дата звернення 17.12.23).

18. 3ds Max 2021 For Beginners: A Tutorial Approach / Prof. Sham Tickoo, 2016. 347 p. URL: https://www.cadcim.com/autodesk-3ds-max-2021-for-beginners-a-tutorial-approach?AspxAutoDetectCookieSupport=1 (дата звернення 20.12.23).

19. Maraffi C. Maya Character Creation: Modeling and Animation Controls, 2018. 101 p. URL: https://books.google.com.ua/books?id=-1GE6YbntJrQC&printsec=copyright&redir_esc=y#v=onepage&q&f=false (дата звернення 20.12.23).

20. Blender For Dummies / Jason van Gumster, 2022. 569 p. URL: https://books.google.com.ua/books/about/Blender_For_Dummies.html?id=gHDJDwAAQBAJ&redir_esc=y (дата звернення 20.12.23).

21. Marchuk A. Renga Architecture: Design, Construction, and Operation, 2005. 702 p.

22. LibreCAD 3 User Guide / LibreCAD Documentation Team, 2023. 124 p.

23.Офіційний сайт Adobe. URL: https://www.adobe.com (дата звернення 17.12.23).

24. Офіційний сайт Terra Tech. URL: https://terratechgame.com (дата звернення 17.12.23).

25. Офіційний сайт Kerbal Space Program. URL: https://www.kerbalspaceprogram.com (дата звернення 17.12.23).

26. Game Engine Architecture / Jason Gregory, 2014. 890 p.

27. Гнучка розробка програмного забезпечення / Jeff Sutherland, 2016. 304 с.

28. -Виявлення та Управління Бізнес-Вимогами / Кеті Сайрант, Вайка Натан, Еллен Готтс. Видавництво «Оріл», 2016. 368 с.

29. IDEF0 Modeling for Organizational Design: A Process Description Capture Method / William F. Tarn. Prentice Hall, 1994. 356 p.

30. Розробка інтерфейсу користувача: Підручник для професіоналів / Деніел Т. Сафаріан, Кеті Шнайдер, 2014. 480 с.

31. Data Flow Diagrams: Simply Put! / Thomas Hathaway, Angela Hathaway, 1998. 220 p.

32. UML Distilled: A Brief Guide to the Standard Object Modeling Language / Martin Fowler, Addison-Wesley, 2003. 208 p.