НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

ННК "Інститут прикладного системного аналізу"
(повна назва інституту/факультету)

Кафедра Системного проектування
(повна назва кафедри)

«До захисту допущено»

Завідувач кафедри
_____ А.І.Петренко
(підпис) (ініціали, прізвище)

"___"_____2017 р.

Дипломна робота

першого (бакалаврського) **рівня вищої освіти**
(першого (бакалаврського), другого (магістерського))

зі спеціальності7.05010102, 8.05010102 Інформаційні технології проектування
7.05010103, 8.05010103 Системне проектування
(код та назва спеціальності)

на тему: Дослідження технологій створення віртуальних лабораторій

Виконав: студент _4__ курсу, групи __ДА-32_
(шифр групи)

Волоха Олександр Олександрович _____
(прізвище, ім'я, по батькові) (підпис)

Керівник доцент, к.т.н., Кисельов Г.Д. _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант економічний _____ доцент к.е.н Рощина Н. В. _____
(назва розділу) (посада, вчене звання, науковий ступінь, прізвище, ініціали) (підпис)

Рецензент _____ _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Нормоконтроль ст. викладач Бритов О.А. _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2017 року

Національний технічний університет України
«Київський політехнічний інститут»

Факультет (інститут) ННК "Інститут прикладного системного аналізу"
(повна назва)

Кафедра Системного проектування
(повна назва)

Рівень вищої освіти Перший(Бакалаврський)
(перший (бакалаврський))

Спеціальність 7.05010102, 8.05010102 Інформаційні технології проектування
7.05010103, 8.05010103 Системне проектування
(код і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ А.І.Петренко
(підпис) (ініціали, прізвище)
«___»_____2017 р.

ЗАВДАННЯ

на дипломний проект (роботу) студенту

Волосі Олександру Олександровичу

(прізвище, ім'я, по батькові)

1. Тема проекту (роботи) Дослідження технологій створення віртуальних лабораторій
керівник проекту (роботи) Кисельов Генадій Дмитрійович, к.т.н., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «___»_____ 20__ р. №_____

2. Строк подання студентом проекту (роботи) 06.06.2017

3. Вихідні дані до проекту (роботи) _____

- Google VR SDK

6. Консультанти розділів проекту (роботи)*

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний	Рощина Н. В., доцент.		

* Консультантом не може бути зазначено керівника дипломного проекту (роботи).

АНОТАЦІЯ

бакалаврської дипломної роботи Волохи Олександра Олександровича

на тему: «Дослідження технологій створення віртуальних лабораторій»

Дипломна робота присвячена дослідженню технологій для розробки віртуальних лабораторій з метою візуалізації процесу математичного моделювання.

У даній роботі ставиться завдання розглянути сучасні інструментальні засоби для створення віртуальних лабораторій. Було проаналізовано найпопулярніші сучасні ігрові рушії, фреймворки та існуючі на ринку рішення.

Повсюдне впровадження дистанційних технологій, що спостерігається в останні роки в усьому світі, диктує необхідність пошуку нових підходів до організації процесів моделювання. Тенденції останніх кількох років в сегменті IT показують зростання створення і поширення різноманітних ресурсів для візуалізації та моделювання тих чи інших процесів, часто вони створюються за принципом хмарних технологій і використовують мережу інтернет. Серед них присутні розробки, які реалізують віртуальні лабораторні роботи.

Результат роботи – порівняльний аналіз та класифікація сучасних популярних інструментальних засобів. Розробка тестового проекту віртуальної лабораторії із використанням технології VR.

Загальний обсяг роботи 82 сторінки, 27 рисунків, 9 таблиць, 9 посилань.

Ключові слова: Віртуальна лабораторія, віртуальна реальність, Ігровий рушій, фреймворк, дистанційне навчання, фізичний рушій.

АННОТАЦИЯ

бакалаврской дипломной работе Волохи Александра Александровича

на тему: «Исследование технологий создания виртуальных лабораторий»

Дипломная работа посвящена исследованию технологий для разработки виртуальных лабораторий с целью визуализации процесса математического моделирования.

В данной работе ставится задача рассмотреть современные инструментальные средства для создания виртуальных лабораторий. Были проанализированы самые популярные современные игровые движки, фреймворки и существующие на рынке решения.

Повсеместное внедрение дистанционных технологий, наблюдается в последние годы во всем мире, диктует необходимость поиска новых подходов к организации процессов моделирования. Тенденции последних нескольких лет в сегменте ИТ показывают рост и распространения различных ресурсов для визуализации и моделирования тех или иных процессов, часто они создаются по принципу облачных технологий и используют сеть интернет. Среди них присутствуют разработки, реализующих виртуальные лабораторные работы.

Результат работы - сравнительный анализ и классификация современных популярных инструментальных средств. Разработка тестового проекта виртуальной лаборатории с использованием технологии VR.

Общий объем работы 82 страницы, 27 рисунков, 9 таблиц, 9 ссылок.

Ключевые слова: Виртуальная лаборатория, виртуальная реальность, Игровой движок, фреймворк, дистанционное обучение, физический движок.

ANNOTATION

On Volokha Oleksander Oleksandrovich bachelor's degree thesis

"Investigation of technologies for creating virtual laboratories"

Thesis is devoted to research technologies for developing virtual laboratories for the purpose of visualizing the process of mathematical modeling.

This paper seeks to examine modern tools to create virtual labs. It analyzes the current most popular game engines, frameworks and existing solutions in the market.

The widespread introduction of distance learning, observed in recent years in the world, requires finding new approaches to modeling processes. Trends in the last few years in the segment of IT show an increase in the creation and distribution of various resources for visualization and simulation of various processes, often they are created on the basis of cloud technologies and use Internet. Among them there are developments that implement virtual labs.

The result - a comparative analysis and classification of modern popular tools. Development of a test project using virtual laboratory technology VR. The delineation of the scope of, and the target audience for specific representatives of tools.

The total volume of 82 pages, 27 figures, 9 tables, 9 bibliographic titles.

Keywords: Virtual Lab, virtual reality, game engine, framework, distance learning, physics engine.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ

VR	Virtyal reality (віртуальна реальність)
ВЛ	Віртуальна лабораторія
СДН	Система дистанційного навчання
GE	Game Engine (ігровий двигун)
БД	База даних
JS	JavaScript (мова програмування)

ВСТУП

Одним з головних напрямків науково-технічного прогресу протягом вже кількох десятиріч є розвиток методів і засобів інформатики та обчислювальної техніки. Використання методів математичного моделювання і комп’ютерного розв’язання інженерних і наукових задач дозволяє значно підвищити ефективність процесів проектування та управління. Впровадження персональних комп’ютерів, комп’ютерних інформаційних мереж, побудова та розвиток інтернету, широке та різноманітне використання методів математичного моделювання привели до розширення як практичної, так і теоретичної баз комп’ютерної математики. Математичне комп’ютерне моделювання стало головним засобом дослідження складних процесів і систем, на якому базуються сучасні підходи до проектування, оптимізації та управління в різних галузях науки і техніки. Обчислювальна математика стала основою для реалізації та комп’ютерного розрахунку методів математичного моделювання.

Методи моделювання в даний час проникали практично в усі сфери людської діяльності: технічні, соціально-економічні, складні економічні, громадські, сфери міжнародних відносин та ін. Це пов'язано з необхідністю розширення і поглиблення знань реального світу. Існує безліч реальних об'єктів і процесів, інформацію про яких ми не можемо отримати через малість або масштабності розмірів (об'єкти мікро-і макрокосмосу); високих або кріогенних температур. Не можемо проводити експерименти - це може бути пов'язано з тривалістю процесу (екологічні); високою вартістю досліджень об'єкта-оригіналу; унікальністю об'єкта дослідження; відсутністю об'єкта-оригіналу (ескізні, архітектурні та конструкторські проекти), небезпекою дослідження (ядерні вибухи) та інші.

Віртуальна лабораторія (ВЛ) – це віртуальна середа, яка дозволяє моделювати поведінку об’єктів реального світу в комп’ютерному середовищі і допомагає в оволодінні новими знаннями та вміннями. Така лабораторія може виступати апаратом досліджень різних природних явищ з можливістю побудови

їх математичних моделей. Використання ВЛ дає змогу не лише спостерігати за певним експериментом, а й безпосередньо брати в ньому участь, а це в свою чергу сприяє засвоєнню знань на більш свідомому та глибокому рівні.

1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Актуальність

Повсюдне впровадження дистанційних технологій, що спостерігається в останні роки в усьому світі, диктує необхідність пошуку нових підходів до організації процесів моделювання. Тенденції останніх кількох років в сегменті IT показують зростання створення і поширення різноманітних ресурсів для візуалізації та моделювання тих чи інших процесів, часто вони створюються за принципом хмарних технологій і використовують мережу інтернет. Серед них присутні розробки, які реалізують віртуальні лабораторні роботи. За деякими з них можна знайти ресурси, які можна використовувати в дистанційному навчанні, бізнесі, проектуванні та багатьох інших галузях. Наприклад, можна привести такі програмні системи як: LabView, Jmcad, Absorb Chemistry (/Electronics / Mathematics / Physics / Advanced Physics), Crocodile Mathematics, Yenka Technology (Science / Mathematics / Programming), Discovery Studio Visualizer, Swiss-PdbViewer, ChemLab, OPNET IT Guru, 3D Human Anatomy, MediView, і багато інших. Таких засобів є величезна кількість, вони охоплюють великий спектр дисциплін. Багато з них є найпотужнішими засобами для проведення віртуальних лабораторних дослідів, бізнес-аналізу, моделювання складних процесів. Однак, в більшості прикладів такі засоби коштують дуже дорого а поріг входження до роботи з програмою є дуже високим і потребує спеціальних знань. Більшість з них дають змогу виконати розрахунки і побудувати моделі але не візуалізовують процеси моделювання достатньо для розуміння того, що відбувається. Також іноді якість реалізації не завжди задовольняє користувача.

1.2 Основні поняття теорії моделювання

Поняття „моделі”, „моделювання”, є основним майже в усіх галузях наукової та інженерної діяльності. Дуже часто через різноманітність напрямків в

окремих теоретичних дисциплінах під моделюванням розуміють суттєво різні теорії, методи та засоби.

1.2.1 Постановка задачі моделювання

Моделювання – це опис певного *об'єкта*. В галузі систем керування об'єктами моделювання є, відповідно, самі об'єкти, системи і процеси керування ними, а також їх складові частини.

Моделювання можна розглядати як *відображення* об'єкта на множину його описів. При цьому, якщо об'єкту O і моделі M властиві певні набори характеристик $O\{X\}$ і $M\{Z\}$ то повинна існувати відповідність характеристик моделі і об'єкта $x_i \leftrightarrow z_j$ Найчастіше модель свідомо будують як спрощений опис об'єкта для полегшення його дослідження. Це не дивно, адже природні об'єкти характеризуються безліччю показників, більшість з яких є несуттєвою з точки зору *мети моделювання*. Так, наприклад, при моделюванні системи керування токарним верстатом несуттєвими характеристиками є його колір, рівень шуму тощо. В результаті між моделлю та об'єктом немає повної відповідності.

Отже, задачу моделювання можна сформулювати таким чином: необхідно для заданого об'єкта підібрати такий опис, який достатньо повною мірою відображав би оригінал з точки зору заданої мети моделювання.

У найпростішому вигляді це відображає рис. 1.1. Як можна побачити з цього рисунка, у зображенні системи використовуються математичні об'єкти:

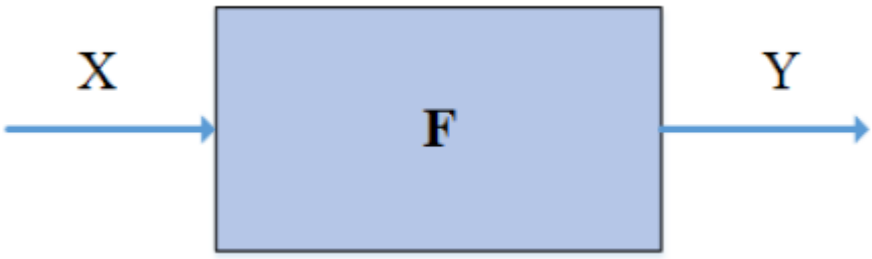

Рисунок 1.1 – Спрощена схема об’єкта моделювання [1]

$$\Theta Y = F[\Theta X], \quad (1.1)$$

$$X = \Theta\, F\, [\Theta Y], \quad (1.2)$$

Формула (1.1) відображає задачу отримання результатів моделювання. Ця задача використовується у більшості задач застосування моделювання: оптимізація процесів, прогнозування характеристик систем, проектування систем управління тощо.

Формула (1.2) відображає задачу знаходження параметрів стану об’єкта за показами вимірювальних приладів, якщо під F розуміти рівняння перетворення цих приладів.

На стадії проектування модель є основою створення системи. За її допомогою розробляють структуру і алгоритм (algorithm) роботи системи, прогнозують її характеристики, розраховують параметри елементів. Результатом цього етапу є проект СУ, який фактично є її моделлю. Процес проектування систем є ітераційною оптимізаційною процедурою поступової деталізації і доповнення комплексу моделей системи.

1.3 Види моделей

Залежно від задач, для розв’язання яких вони призначені, використовуються дуже різні моделі. Загальна схема видів моделей представлена на рис. 1.2.

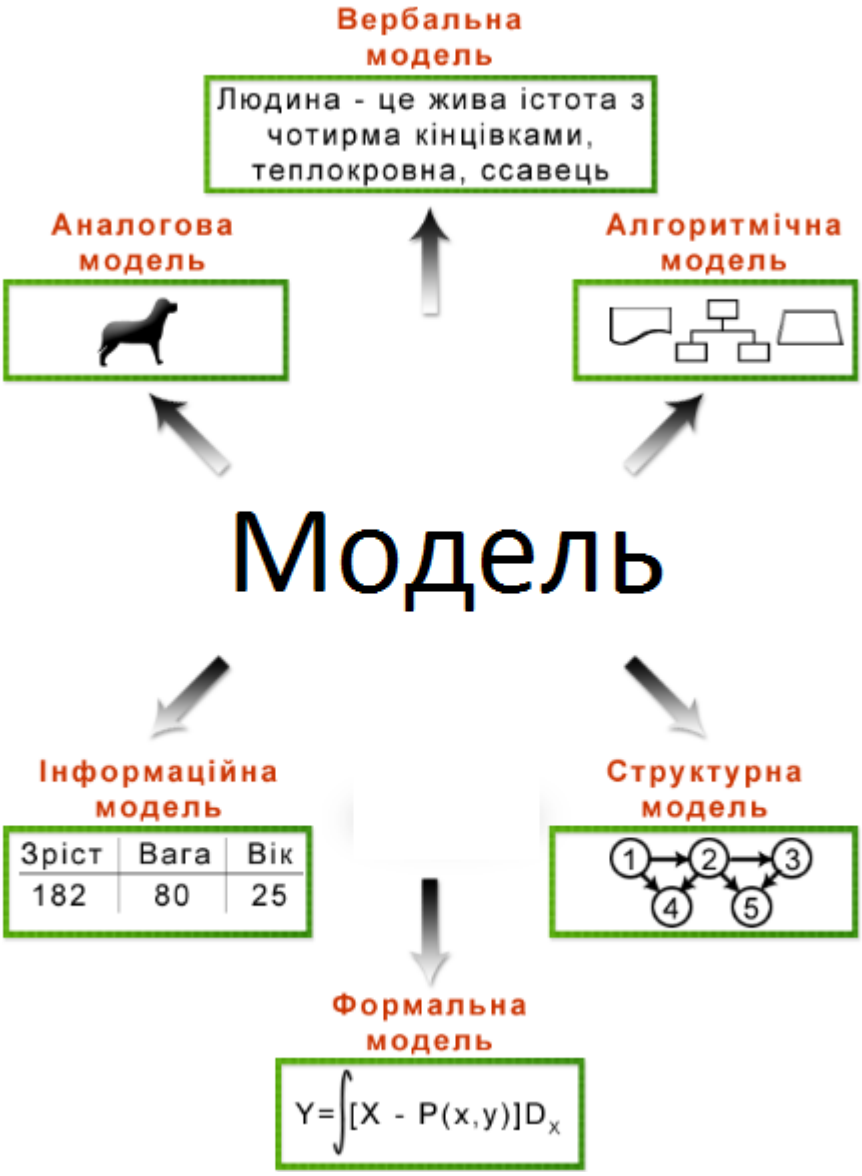

Рисунок 1.2 – Схема види моделей [1]

1.3.1 Моделі вербальні, формальні, алгоритмічні, графічні, фізичні

Залежно від способу опису об’єкта-оригінала моделі розділяються на нижчевказані:

Вербальні моделі (verbal model) використовують словесний опис об’єкта. Такі моделі часто використовують в нетехнічних галузях, а також на початковому етапі моделювання в техніці.

Формальні моделі (formal model) використовують опис об’єкта моделювання у вигляді формул і подаються системою математичних співвідношень.

Алгоритмічні моделі (algorithmic model) подають об’єкт у вигляді послідовності дій, які дозволяють отримати його необхідну характеристику.

Графічна модель (graphical model) зображує модель у наочному вигляді. До графічних моделей відносять різноманітні схеми, які складають конструкторську документацію (електричні, гідравлічні, пневматичні, механічні, схеми програм і даних тощо), графіки, креслення форми об’єктів у різник проекціях, плани і карти тощо. З розвитком комп’ютерної графіки та її застосування для комп’ютерної анімації, симуляції і ігор графічні моделі за масштабом застосування вийшли на перші позиції і наразі поступово створюють віртуальний світ.

Фізична модель (physical model) подає об’єкт-оригінал іншим об’єктом такої ж фізичної природи (масштабні моделі) або іншої (аналогові моделі). Основою фізичного моделювання є теорія подібності. Фізичне моделювання застосовують при дослідженні систем, для яких вихідні дані відомі з обмеженою точністю чи неможливо дати точний математичний опис їх функціонування, а отримання експериментальних характеристик пов’язано з надмірними труднощами та витратами.

В аналогових моделях (analog model) фізична природа моделі і об’єкта різні, а їх математичні описи подібні і, крім того, подібні рівняння, які описують їх окремі елементи. В моделі-аналозі реакції на збурення подібні до реакцій на аналогічні збурення об’єкта. Моделі-аналоги складаються з окремих блоків, які моделюють фізичні елементи, а не з блоків, які виконують окремі математичні операції. Кожному фізичному параметру в об’єкті однозначно відповідає деякий елемент в моделі-аналозі. Найчастіше використовують електронні моделі при дослідженні поведінки систем, конструювання та безпосереднє вивчення яких пов’язано з надмірними труднощами та витратами.

Масштабна модель (scale model) – це аналогова модель, в якій між параметрами об’єкта і моделі однакової фізичної природи існує однозначна відповідність, а також відповідність між впливами та реакцією на них. В масштабній моделі кожен елемент в масштабі повторює відповідний елемент об’єкта.

1.3.2 Моделі геометричні, структурні, функціональні, інформаційні

Залежно від властивостей об’єктів, які відображають моделі виділяють:

– геометричні моделі;

– структурні моделі;

– функціональні моделі;

– інформаційні моделі.

Геометричні моделі (geometric model) відображають форму та розташування об’єкта моделювання та його складових частин. Геометричними моделями є різноманітні креслення механізмів, будівель тощо.

Структурна модель (structural model) подає об’єкт моделювання з точки зору його складу та взаємозв’язку частин (елементів системи) між собою та з зовнішнім середовищем. Зв’язки між елементами можуть бути: фізичні; логічні; інформаційні. Сукупність елементів системи та зв’язків між ними утворюють структуру системи. Структурні моделі найчастіше існують у формі різноманітних структурних та принципових схем.

Функціональні моделі (functional model) описують процеси, що відбуваються в об’єкті моделювання.

Інформаційна модель (information model) – система даних про об’єкт та опис потоків даних в процесі його функціонування.

1.3.3 Моделі статики, моделі динаміки

Залежно від наявності відображення змін стану об’єкта у часі моделі поділяються на:

– моделі статики;

– моделі динаміки.

Моделі статики (static model) відображають стан та функціонування об’єкта без урахування їх змін у часі. Як правило, вони подаються у вигляді функціональних залежностей, рівнянь та систем рівнянь.

Моделі динаміки (dynamic model) відображають поведінку об’єкта у часі. Моделі динаміки багатші за моделі статики, оскільки останні можуть розглядатися як окремий випадок для певного фіксованого моменту часу. Відповідно і форм подання моделей динаміки значно більше (диференціальні рівняння, операторні рівняння, спектральні подання тощо).

1.3.4 Моделі статики, моделі динаміки

Розглядаючи види моделей, ще раз повернемося до рис. 1.1. Цей рисунок демонструє взаємозв’язок трьох базових моделей: моделі вхідних впливів X, моделі системи F, моделі станів і вихідних сигналів Y. Якщо ці моделі подані як моделі динаміки, тобто зображають поведінку вхідних впливів, моделі системи, станів і вихідних сигналів у часі, то вони є моделями певних процесів. Система керування здійснює перетворення вхідних впливів на зміни станів, отже модель функціонування системи є моделлю перетворення.

Система керування є складним технічним об’єктом, який може розглядатися у різних аспектах: склад і структура, конструкція (в тому числі форма і розташування окремих частин системи), функції системи (в тому числі алгоритм її функціонування), параметри окремих блоків і системи в цілому. Отже модель системи є взаємопов’язаним комплексом моделей структури, функціонування, розташування в різноманітних формах (алгоритмічній, інформаційній, формальній, графічній тощо).

1.3.5 Моделі детерміновані, стохастичні, нечіткі, узагальнені

За степенем та характером невизначеності моделі поділяються на:

– детерміновані;

– стохастичні;

– нечіткі;

– узагальнені.

Детерміновані моделі (deterministic model) не враховують можливі випадкові відхилення характеристик об’єкта і вхідних впливів від номінальних значень.

Стохастичні моделі (stochastic model) розглядають поведінку системи в умовах дії випадкових впливів та випадкової зміни параметрів системи. Інколи розглядають також випадкові зміни структури системи, зумовлені ненадійністю зв’язків між підсистемами та іншими причинами.

Нечіткі моделі (fuzzy model) використовують у випадках, коли окремі параметри системи задані експертом з кінцевим ступенем впевненості.

Узагальнені моделі (generalized model) використовуються при моделюванні систем, в яких частина параметрів задана достовірно, частина отримана в результаті статистичної обробки певних випадкових процесів, а частина задана експертним методом[1].

1.3.6 Моделі агрегатні, комплексні

Залежно від способу представлення складного об’єкта (системи) розрізняють моделі:

– агрегатні;

– комплексні.

Агрегатна модель (aggregate model) складної системи складається з моделей окремих підсистем та опису їх взаємодії (рис. 1.3, б). Якщо розглядати

для прикладу деяку систему керування, то моделі підсистем подаються окремими рівняннями, що зв’язують вихідні сигнали з вхідними і параметрами підсистеми, а агрегатна модель буде подаватися системою цих рівнянь.

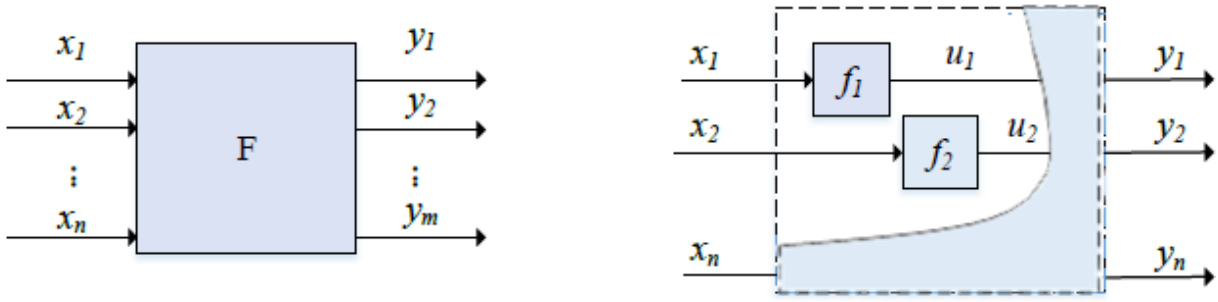

Рисунок 1.3 – Декомпозиція комплексної моделі (а) на агрегати (б) [2]

Комплексна модель (complex model) подає систему в цілому, не розділяючи її на підсистеми і окремі внутрішні процеси (рис. 1.3, а). Комплексна модель може бути отримана з агрегатної шляхом зведення системи рівнянь до одного рівняння, що зв’язує вхідні і вихідні сигнали системи, методом підстановки.

Перевага агрегатної моделі у більшій простоті її отримання, оскільки моделі окремих підсистем простіші за модель системи в цілому. Перевага комплексної моделі у тому, що для її отримання немає необхідності досліджувати внутрішню структуру системи.

1.3.7 Моделі аналітичні, імітаційні

Залежно від способу отримання результатів моделювання розрізняють математичні моделі:

– аналітичні;

– чисельні;

– імітаційні.

Аналітичне моделювання (analytical modeling) – знаходження характеристик об’єкта на основі формальної або алгоритмічної моделі шляхом виконання певних математичних перетворень: розв’язання рівнянь та систем рівнянь тощо.

Чисельне моделювання – знаходження характеристик об’єкта на основі формальної або алгоритмічної моделі шляхом застосування числових методів до розв’язання рівнянь та систем рівнянь тощо.

Імітаційне моделювання (simulation) – проведення на ЕОМ чисельних експериментів з математичною моделлю, що описує поводження складної системи протягом певного періоду часу. Застосовується, як правило, в тих випадках, коли аналітичні способи дослідження моделі відсутні, а їх пошук потребує дуже великих витрат. Алгоритми імітаційного моделювання можуть враховувати як детермінованість, так і стохастичність, зв’язки і залежності, що характеризують об’єкт моделювання. Найбільше розповсюдження отримали стохастичні методи імітаційного моделювання, оскільки для більшості складних систем відомі лише усереднені значення параметрів. Оскільки імітаційне моделювання являє собою експеримент, важливе значення має застосування методів планування експерименту. Основні проблеми, які доводиться розв’язувати в зв’язку з цим:

1) забезпечення стохастичної збіжності;

2) намагання зменшити кількість різних комбінацій факторів, що впливають на об’єкт, без зменшення кількості отриманої інформації;

3) вибір плану експерименту, який залежить від глибини розуміння експериментатором суті процесів, що відбуваються в системі.

1.4 Характеристики моделей

Ефективне використання моделей можливе лише за умови, що їх характеристики відповідають певним вимогам. Основними характеристиками моделей є точність, вірогідність, адекватність, складність, універсальність.

1.4.1 Точність

Точність математичної моделі (accuracy of mathematical model) – її властивість, яка відбиває ступінь збігу передбачених з її допомогою значень характеристик об’єкта з дійсними значеннями цих характеристик. За дійсне значення характеристики об’єкта звичайно приймають експериментально отримані значення або достовірно відомі факти.

1.4.2 Вірогідність

Вірогідністю характеризуються моделі, для яких не визначено метрику.

Вірогідність (reliability of mathematical model) – це ймовірність відсутності помилки при побудові моделі.

$$P0 = 1 - Pпом. \quad (1.3)$$

Ймовірність помилки розраховується на основі аналізу усіх можливих джерел помилок. Але слід брати до уваги, що у деяких задачах локальна помилка може не привести до загальної помилки моделювання, наприклад у задачах знаходження мінімальних або максимальних шляхів за допомогою структурних моделей у вигляді графів, якщо цей шлях не проходить через помилково визначене ребро.

1.4.3 Адекватність

Необхідна умова для переходу від дослідження об’єкта до дослідження моделі і подальшого перенесення результатів на об’єкт дослідження – вимога адекватності моделі і об’єкта.

Адекватність (adequacy of mathematical model) – це правильне відтворення моделлю з необхідною повнотою всіх властивостей об’єкта, важливих для цілей даного дослідження.

Будь-яка система чи підсистема може бути подана різними способами, які значно відрізняються один від одного за складністю і деталізацією. В більшості випадків в результаті дослідження з'являється декілька різних моделей одної і тої ж системи. При цьому, залежно від глибини аналізу прості моделі послідовно заміняються все більш складними.

1.4.4 Складність

Вже відзначалося, що будь-яка модель є спрощеним описом об'єкта. Складність моделі є комплексною характеристикою, яка переважно сприймається інтуїтивно. Залежно від виду моделі розрізняють декілька видів складності.

Структурна складність визначається кількістю елементів, зв'язків між ними та показником нерегулярності цих зв'язків.

Функціональна складність визначається кількістю вхідних і вихідних даних, обчислювальною складністю моделі.

Найглибше пророблені методи визначення складності для алгоритмічних моделей. Обчислювальна складність алгоритму – поняття в інформатиці та теорії алгоритмів, що позначає функцію залежності обсягу роботи, виконуваної деяким алгоритмом. При оцінюванні складності алгоритмів враховують кількість вхідних і вихідних даних, кількість операцій, кількість циклів і викликів зовнішніх функцій тощо.

1.4.4 Універсальність

Ступінь універсальності математичної моделі визначається її застосуванням до аналізу чисельної групи однотипних об'єктів, до їх аналізу в одному чи багатьох режимах функціонування.

Як приклад найуніверсальніших моделей можна навести модель гравітаційної взаємодії («закон всесвітнього тяжіння» Ньютона). Меншу

універсальність мають потокові моделі (закони Кірхгофа) – вони справедливі лише для лінійних систем. Ще вужче застосування мають моделі конкретних об’єктів – їх універсальність проявляється при розгляді об’єкта за різноманітних умов (різних вхідних даних, впливів тощо). Підсумовуючи сказане, слід зазначити, що математичне моделювання та ідентифікації об’єктів і систем є складним багатоетапним процесом, який передбачає створення моделі, виконання розрахунків відповідно до моделі та використання отриманих результатів. І на кожному етапі крім формальних математичних методів необхідний ще й досвід інженера, дослідника.

При цьому слід брати до уваги вплив точності і складності не тільки на перший етап, а й, що є найголовнішим, на остаточний результат застосування моделі. Адже дуже часто висока точність моделі зводиться нанівець похибками розрахунків при застосуванні моделі в задачах оптимізації, оцінювання тощо через велику складність. І тільки значний досвід дослідника дозволяє вже з самого початку моделювання досягти цього компромісу[1].

1.5 Висновок

В даному розділі було розглянуто основні види моделей та їхні характеристики. Розуміння основ і методів математичного та комп'ютерного моделювання є необхідним при створенні та використанні об’єктів віртуальної лабораторії. Дуже важливо правильно і максимально точно описати модель на етапі проектування

Оскільки моделювання – це досить складний процес і для кращого сприйняття та засвоєння тих чи інших методів, алгоритмів або рішень для конкретних задач ми дуже часто потребуємо візуалізацію окремих етапів або всього процесу.

2 ВІРТУАЛЬНІ ЛАБОРАТОРІЇ

Як було зазначено раніше ВЛ – це комплекс програм або програмно-апаратний засіб, а також набір документації по їх використанню, що дозволяють проводити експеримент повністю або частково на математичній моделі.

2.1 Призначення

Найбільшу популярність віртуальні лабораторії знайшли у сфері навчання. Але також успішно використовуються в науці та техніці зокрема для підготовки фахівців або для проведення неможливих у реальному житті експериментів.

Віртуальні лабораторні роботи можуть використовуватися як у навчальних закладах, так і в навчальних центрах різних організацій. Такі лабораторні роботи значно підвищують ефективність навчального процесу і надають широкі можливості для формування та вдосконалення професійних навичок та інтуїції, а також розвивають творчі здібності.

Віртуальні лабораторні роботи дають можливість отримувати реальні умови для виконання експериментальних завдань, порівнювати виміряні дані експерименту із сучасним фізичним експериментом, який проведено на дорогому науков-дослідному обладнанні, і таким чином засвоювати нові інформаційні технології. Прикладом таких віртуальних робіт є система LabVIEW (Laboratory Virtual Instrumentation Engineering Workbench) компанії National Instruments, яка дає можливість створювати вимірювальні комплекси й системи автоматизації керування на базі віртуальних приладів.

Поєднання віртуальної і реальної дійсності змушує людину широко застосовувати довідкову та наукову літературу, привчає самостійно мислити і приймати рішення, стимулює до самоосвіти і дозволяє розкрити їхні творчі можливості. Моделювання електронних пристроїв в комп'ютерному класі або вдома і візуалізація результатів у вигляді осцилограм, графіків, характеристик, показань віртуальних приладів сприяє кращому розумінню принципів функціонування реальних схем управління і контролю технологічними

процесами виробництва. Експерименти на моделях доповнюють і розширюють реальні фізичні експерименти, дозволяють досліджувати аварійні режими, неприпустимі при натуральних випробувальних пристроях, уповільнити або прискорити розвиток процесів в пристроях, що дозволяє більш глибоко засвоїти принципи їх роботи. Також слід зазначити економічну ефективність застосування імітаційно-моделюючих програмних засобів. Робота в віртуальної лабораторії дозволяє без великих матеріальних затрат довести до кінця будь-які рішення, вибрати оптимальний шлях, а вже потім втілювати його в життя. Крім того, зміна поколінь електронних компонентів відбувається дуже швидко і вдосконалення фізичної бази лабораторій відстає від реального життя.

2.2 Моделювання у віртуальних лабораторіях

Створення комп'ютерних імітаційних моделей починається з початкової ідеалізації представлення про об'єкт і створення на його основі першого варіанта моделі. Робота з моделлю дає можливість з'ясувати, якої інформації про об'єкт бракує, а яка вимагає уточнення. На основі отриманих даних будується програма наступних емпіричних досліджень об'єкта, результати яких допомагають побудувати інший, уточнений варіант його моделі. За необхідності інтеграційні цикли можуть повторюватися кілька разів. Перевага комп'ютерного моделювання полягає в наявності такої імітаційної моделі, що заміщає природний експеримент над самим об'єктом, дає можливість замінити його модельним експериментом, у якому модель імітує поводження об'єкта при різних початкових даних, вихідних параметрах і обмеженнях.

Поведінкою та властивістю об'єктів моделювання як правило керують скріпти, а обробка результатів взаємодії об'єктів відбувається на сервері і відсилається клієнту після обрахування.

2.3 Професійні рішення

Сьогодні на ринку існує багато професійних рішень для моделювання складних систем і процесів. Більшість із них мають можливість візуалізовувати

процесс роботи, а також мають багато сторонніх бібліотек і готових рішень. В даному підрозділі розглядаються декілька найпотужніших і найпопулярніших рішень.

2.3.1 MatLAB

MatLab містить обчислення, візуалізацію і програмування в зручному середовищі, де задачі і розв'язки виражаються у формі, близькій до математичної.

Типові використання MatLab:

– математичні обчислення;

– створення алгоритмів;

– моделювання;

– аналіз даних, дослідження і візуалізація;

– наукова й інженерна графіка;

– розробка застосувань, включаючи створення графічного інтерфейсу.

MatLab – інтерактивна система, в якій основним елементом даних є масив. Це дозволяє розв'язувати різні задачі, пов'язані з технічними обчисленнями, особливо ті, в яких використовуються матриці і вектори, у кілька разів швидше, ніж при написанні програм з використанням таких «скалярних» мов програмування, як C.

У MatLab важлива роль приділяється спеціалізованим групам програм, які називаються toolboxes. Вони дозволяють застосовувати спеціалізовані методи.

Toolboxes – це колекції функцій MatLab (M-файлів), що дозволяють розв'язувати окремі класи задач. Toolboxes застосовуються для обробки сигналів, моделювання систем контролю, нейронних мереж, нечіткої логіки тощо.

Система MatLab складається з п'яти основних частин.

1. Мова MatLab. Це мова матриць і масивів високого рівня з керуванням потоками, функціями, структурами даних, введенням-виведенням.

2. Середовище MatLab. Це набір інструментів і пристосувань, з якими працює користувач чи програміст MatLab. Воно містить у собі засоби для керування змінними в робочому просторі MatLab, введення і виведення даних, а також для створення, контролю і налагодження M-файлів і застосувань MatLab.

3. Керована графіка. Це графічна система MatLab, що містить у собі команди високого рівня для візуалізації дво- і тривимірних даних, обробки зображень, анімації й ілюстративної графіки. Вона також містить у собі команди низького рівня, що дозволяють редагувати зовнішній вигляд графіки.

4. Бібліотека математичних функцій. Це велика колекція обчислювальних алгоритмів від таких елементарних функцій, як сума, синус, косинус, комплексна арифметика, до більш складних, таких як обернення матриць, знаходження власних значень, функції Бесселя, швидке перетворення Фур'є тощо.

5. Програмний інтерфейс. Це бібліотека, що дозволяє писати програми на Сі, які взаємодіють з MatLab. Вона містить засоби для виклику програм з MatLab (динамічний зв'язок), використовуючи MatLab як обчислювальний інструмент, і для читання-запису M-файлів.

Simulink – це інтерактивна система для моделювання нелінійних динамічних систем, що супроводжує MatLab. Вона являє собою графічне середовище, що дозволяє створювати модель процесу шляхом перетаскування блоків і діаграм на екрані і маніпуляції ними. Simulink працює з лінійними, нелінійними, неперервними, дискретними, багатовимірними системами.

Blocksets – це доповнення до Simulink, що забезпечують бібліотеки блоків для таких спеціалізованих застосувань, як зв'язок, обробка сигналів, енергетичні системи.

Real-Time Workshop – це програма, що дозволяє генерувати C-код з блоків діаграм і запускати його на виконання на різних системах реального часу.

Операційне середовище системи MatLab – це інтерфейс, що підтримує зв'язок цієї системи із зовнішнім світом – діалог з користувачем через командний рядок чи графічний інтерфейс, перегляд робочої області і шляхів доступу, редактор і відлагоджувальник M-файлів, робота з файлами й оболонкою DOS,

експорт і імпорт даних, інтерактивний доступ до довідкової інформації, динамічна взаємодія із зовнішніми системами Microsoft Word, Excel Microsoft Word, Excel тощо. Реалізуються інтерфейси через командне вікно, інструментальну панель, системи перегляду робочої області і шляхів доступу, редактор М-файлів, спеціальні меню і т. д[2].

2.3.2 LabView

LabVIEW - це кроссплатформенная графічне середовище розробки додатків. І хоча цей продукт часом тісно пов'язаний з апаратним забезпеченням National Instruments, він тим не менш не пов'язаний з конкретною машиною. Існують версії для Windows, Linux, MacOS. Програми будуть виглядати однаково в усіх системах. Код, згенерований LabVIEW також може бути також виконаний на Windows Mobile або PalmOS (справедливості заради треба відзначити, що підтримка PalmOS припинена). Ця мова може з успіхом використовуватися для створення великих систем, для обробки текстів, зображень і роботи з базами даних.

LabVIEW має вельми високорівневу мову програмування. Однак ніщо не заважає включати «низькорівневі» модулі в LabVIEW-програми. Навіть якщо ви хочете використовувати асемблерні вставки - це теж можливо, треба лише згенерувати DLL і вставити виклики в код. З іншого боку, високорівнева мова дозволяє запросто виробляти досить нетривіальні операції з даними, на які в звичайній мові могли піти багато рядки (якщо не десятки рядків) коду. Втім, заради справедливості треба зазначити, що деякі операції низькорівневих мов (наприклад, роботу з покажчиками), не так просто реалізувати в LabVIEW зважаючи на його «високорівневість». Зрозуміло, мова LabVIEW включає основні конструкції управління, що мають аналоги і в «традиційних» мовах:

- змінні (локальні або глобальні).
- розгалуження (case structure).
- For - цикли з перевіркою завершення і без.
- While – цикли.

- Угруповання операцій.

2.4 Проекти університетів

Віртуальні лабораторії досить часто застосовуються для навчання у провідних університетах світу. Деякі з них навіть створюють окремі підрозділи для роботи з віртуальними лабораторіями.

2.4.1 Star(MIT)

STAR (Software Tools for Academics and Researchers) - програма Массачусетського технологічного інституту (MIT) з розробки віртуальних лабораторій для досліджень і навчання. Діяльність програми полягає в розробці навчальних і дослідницьких програм із загальної біології, біохімії, генетики, гідрології, в області розподілених обчислень. Більшість додатків реалізовані в java або в html.

- StarBiochem - 3D-визуализатор молекул білків. Має гнучке і детальне налаштування.

- StarGenetics. - дозволяє моделювати процеси схрещування, вивчати закономірності успадкування моногенних ознак (т.зв. закони Менделя).

- StarORF. - дозволяє навчитися ідентифікувати так звані відкриті рамки зчитування (англ - ORF - Open Reading Frame) - одиниці в складі ланцюга ДНК або РНК, здатні кодувати білок.

- StarMolSim - це серія інструментів, що моделює процеси молекулярної динаміки. Кожен з інструментів має широкий набір вхідних значень і, аналогічно, широкий набір вихідних значень для аналізу і дослідження[5].

2.4.2 PhET

Проект розроблений Університетом Колорадо. Проект включає безліч віртуальних лабораторій, які демонструють різні явищ в області фізики, біології, хімії, математики, наук про Землю. Досліди мають високу пізнавальну цінність і

при цьому дуже цікаві.

2.5 Висновки

На сьогоднішній день існує багато чудових професійних продуктів, розроблених спеціалістами з найкращих компаній та науковців з провідних університетів світу. Деякі проекти призначеня для навчання зокрема дистанційного, а деякі успішно використовуються в сферах машино та авіа-будування, проектуванні інтегральних схем та інших. Але у кожного з пакетів існують певні недоліки. Наприклад дуже велика ціна, або не підтримка нових технологій(наприклад віртуальної реальності).

3 ТЕХНОЛОГІЇ ДЛЯ СТВОРЕННЯ ВІРТУАЛЬНИХ ЛАБОРАТОРІЙ

3.1 Що необхідно для створення

Процесс створення віртуальної лабораторії за своїм видом дуже схожий на процесс створення звичайного ПЗ, але звичайно має певні особливості. Є певні необхідні інструменти і обмеження, які обов'язково повинна мати середа або фреймворк для можливості реалізації проекту ВЛ. По-перше це фізичний двигун. Виконувати обрахунки вручну, звичайно можна, але оскільки лабораторія – це комплекс різних об'єктів, реалізувати фізичні підрахунки без затримок буде досить складним завданням і взагалі це того не вартує. Оскільки нижче приведені інструменти будуть мати вбудовані фізичні двигуни, написані такими компаніями, як наприклад Nvidia. Було б цікаво, якщо інструментарій дозволяв би працювати з VR та мати свіжі версії Google VR SDK. Вибір мови програмування не є критичним. Також на вибір буде впливати вартість і доступність технології.

Основні критерії в виборі технології:

- Можливість налаштувати інтернет взаємодію
- Підтримка високорівневих мов програмування
- Основними вимогами до технології є:
- Інтуїтивність, зручність інтерфейсу IDE
- VR SDK
- Кросплатформеність
- Фізичний двигун
- Інструменти для реалізації мережевої взаємодії
- Мова програмування
- Додаткові сервіси

3.2 Які інструменти використовували раніше

В даному пункті розглянуто основні технології, що використовувались раніше та вже майже втратили свою актуальність.

3.2.1 CGI

Застаріла технологія для обробки даних зі сторони сервера. Хоча десять років тому вона була базовим засобом для роботи в мережі інтернет. Основою технології є передача на ПК користувача HTML документу з полями для вводу інформації та пунктами вибору. Після заповнення документа він відсилається на сервер і обробляється за допомогою спеціального серверного програмного забезпечення. Недоліком є сильне навантаження серверу, велика затримка на дії користувача, а також неможливість перевірки введених даних безпосередньо на комп'ютері користувача. Також відсутня можливість керування графікою. За допомогою цієї технології раніше створювали віртуальні тести із завданнями у вигляді малюнків.

3.2.1 FLASH

Macromedia Flash і Macromedia Shockwave – це технології, які поєднують в собі роботу із векторною графікою, відео, звуком і можливістю інтерактивного керування. Вони є платформонезалежними – файли можуть бути запущені під різними операційними системами за наявністю на комп'ютері вільно завантажуємого програми-програвача. Об'єми даних, що передаються по мережі не надто великі через використання внутрішньої компресії. Одною із переваг технології Flash є можливість супроводження моделей звуком. Засоби розраховані на створення анімацій та мультиплікацій і для цього підходять ідеально, але якщо необхідні більш складні математичні розрахунки, то програмування стає досить важким, якщо взагалі можливим. Справа в тому, що мова програмування ActionScript не була розрахована на таку область застосування.

3.3 Інтернет лабораторії

Стрімкий розвиток веб-технологій посприяв створенню цікавих проектів безпосередньо на сторінках веб-сайтів.

3.3.1 JavaScript

Мова JS органічно пов’язана з HTML. Зараз вона є основним засобом для обробки дій користувача без відправки даних на сервер після кожної дії. Можна дуже легко працювати з графікою та навіть симулювати певні фізичні явища та описувати моделі за допомогою основних можливостей мови. Якщо мова буде йти про певну перевірку знань, то цей підхід має суттєвий недолік, тому що будь-яка людина може відкрити вихідний код сторінки і подивитись інформацію, яку розробник не хотів показувати користувачеві.

3.3 Ігрові рушії

Для створення віртуальної лабораторії чудово підійдуть технології пов’язані з ігровою розробкою. Адже вони, як правило мають власні фізичні двигуни, пакети з плагінами віртуальної реальності, а також зручну IDE і досить простий і інтуїтивно зрозумілий інтерфейс.

3.3.2 Unreal Engine

Unreal Engine - Написаний мовою C++, рушій дозволяє створювати ігри для більшості операційних систем і платформ: Microsoft Windows, Linux, Mac OS і Mac OS X, консолей Xbox, Xbox 360, PlayStation 2, PlayStation Portable, PlayStation 3, Wii, Dreamcast і Nintendo GameCube. У березні 2010 робота рушія була продемонстрована на комунікаторі Palm Pre, що базується на мобільній платформі webOS.

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 (тільки для Xbox360, PlayStation 3,

Nintendo Wii і Microsoft Windows), модулі для роботи з мережею й підтримка різних пристроїв вводу.

Для гри у мережі підтримуються технології Windows Live, Xbox Live, і GameSpy, включаючи 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. Один з найвідоміших представників жанру, Lineage II, використовує рушій Unreal Engine.

5 листопада 2009 року був випущений пакет Unreal Development Kit, безкоштовна версія Unreal Engine для некомерційного використання з можливістю купівлі дешевої комерційної ліцензії. Інтерфейс двигуна представлений на рис. 3.1.

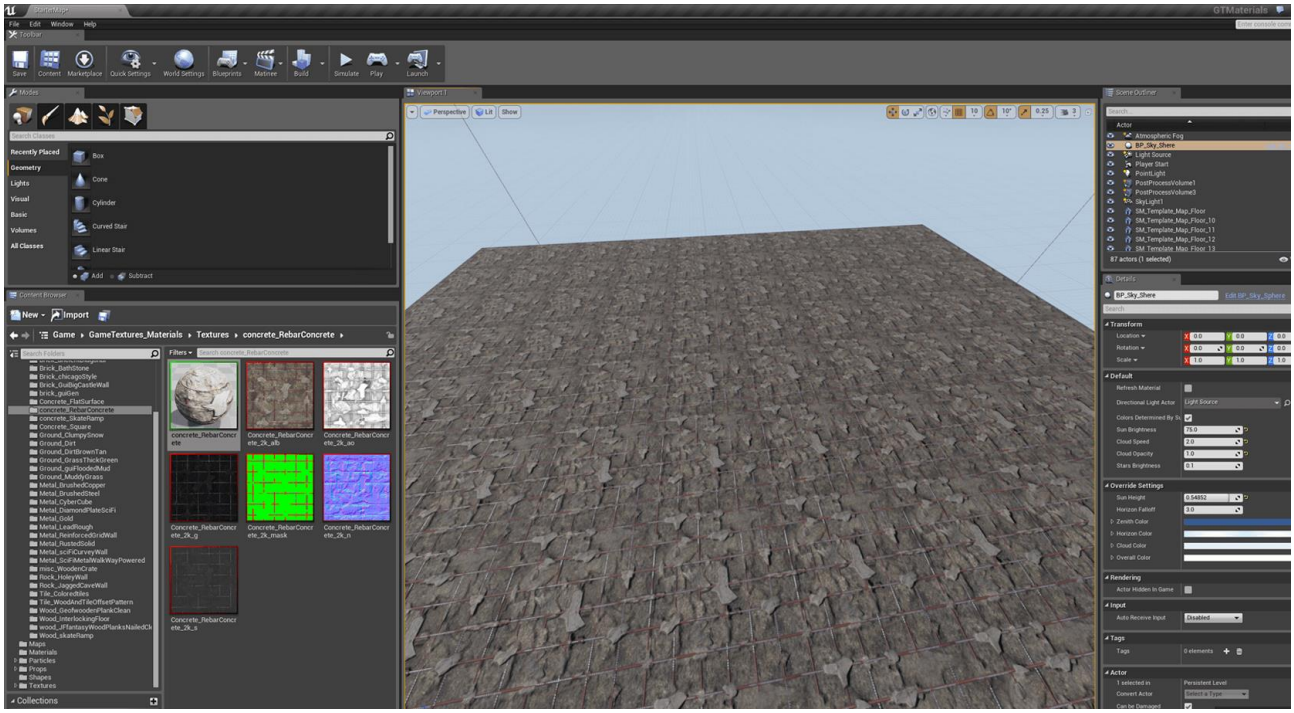

Рисунок 3.1 - Інтерфейс ігрового двигуна UE4.

Для описання логіки гри використовується C++ подібний UnrealScript. Усі елементи ігрового рушія представлені у вигляді об'єктів, що мають набір

характеристик, і клас, який визначає доступні характеристики. У свою чергу будь-який клас є «дочірнім» класом object. Серед основних класів і об'єктів можна виділити наступні:

Актор (actor) — базовий клас, що містить усі об'єкти, які мають відношення до ігрового процесу й мають просторові координати.

Пішак (pawn) — фізична модель гравця або об'єкта, керованого штучним інтелектом. Метод керування описаний спеціальним об'єктом, такий об'єкт називається контролером. Контролер штучного інтелекту описує лише загальну поведінку пішака під час ігрового процесу, а такі параметри як «здоров'я» (кількість пошкоджень, після яких пішак перестає функціонувати) або, наприклад, відстань, на якій пішак звертає увагу на звуки, задаються для кожного об'єкта окремо.

Світ, рівень (world, game level) — об'єкт, що характеризує загальні властивості «простору», наприклад, силу тяжіння й туман, у якому розташовуються всі актори. Також може містити в собі параметри ігрового процесу, як, наприклад, ігровий режим, для якого призначений рівень[6].

3.3.2 Shiva 3D

ShiVa3D - тривимірний ігровий движок з графічним редактором, призначеним для створення додатків та ігор для веб, консолей і мобільних пристроїв.

За допомогою Шиви можна робити програм та ігри для Windows, Linux, Mac OS, iOS, Android, Palm OS і Wii. Так само є плагін для перегляду 3D прямо в браузері. ShiVa3D складається з чотирьох частин: Редактор, Ігровий движок, Інструмент розробника (програма для складання проектів) і Сервер. Ігровий движок Шиви базується на Open GL або DirectX графіці і фізиці ODE.

Движок вміє малювати ландшафт, океан і різні тривимірні моделі з використанням шейдеров. Є статичне і динамічне освітлення і тіні, динамічні частинки, різні ефекти, анімації, елементи призначеного для користувача

інтерфейсу, можливість створення багатокористувацьких ігор і відтворення звуків.

Движок розширюється за допомогою плагінів, можна наприклад, замінити ODE фізику на PhysX. Вбудований WYSIYWG редактор дозволяє створювати ігри та програми з використанням всіх можливостей движка. Для програмування в основному використовується Lua, але можна писати оптимізувати скрипти на C ++. Редактор має 4 редакції: PLE (free), Basic (€ 169), Advanced (€ 1499), Educational (free). Чим краще редакція, тим більше переваг є. Зараз редактор працює тільки під Windows, але в наступній версії обіцяють кроссплатформенность. Інтерфейс двигуна представлений на рис. 3.2.

Рисунок 3.2 - ShiVa3D.

3.3.3 Unity

Для відображення моделей, процесів і явищ використовується візуальне представлення у вигляді 3D графіки. Технологічною основою уявлення є графічний інструмент для розробки тривимірних додатків Unity3D. Unity3D - це багатоплатформовий движок для розробки інтерактивних додатків з графікою, що відтворюється в реальному часі. Цей графічний движок найбільш поширений серед розробників тривимірних великомасштабних ігор. Движок має власний редактор, розробка продуктів ведеться за допомогою мови C#, що

дозволяє створювати додатки, що описують складні фізичні процеси. Також цьому сприяє високий рівень абстракції програмного інтерфейсу. Процес розробки 3-мірних середовищ об'єктно-орієнтована, тобто побудова середовища поділяється на об'єкти з поведінкою. Unity3D підтримує велику кількість апаратних платформ[4]. Створений на основі мови C ++, що робить движок швидким і продуктивним. Цей движок задовольняє ряду поставлених до нього вимог, а саме:

- кінцевий продукт є мультимедійним 3D об'єктом, вбудованим в HTML-сторінку;

- кінцевий продукт є об'єктом високого рівня абстракції прототипів об'єктів;

- забезпечення високої якості графічного представлення інформації;

- бібліотека 3D об'єктів має можливість працювати з сучасними форматами тривимірної графіки - * .3ds, * .dae, * .fbx, * .flt;

- підтримка мов високого рівня (C ++, C #, Java) для забезпечення ефективного процесу розробки;

- наявність ліцензії для вільного використання в некомерційних цілях;

- наявність редактора програмної і графічної розробки об'єктів;

- можливість підключення сторонніх бібліотек об'єктів (бібліотеки для обробки даних, веб-сервіси, драйвери баз даних і т.п.).

Програмна бібліотека мультимедійних 3D об'єктів уніфікує процес створення і обробки віртуальних лабораторних робіт, вона надає необхідні типи даних і механізми обробки створених віртуальних лабораторних робіт. Програмний засіб Unity3D надає можливість створення плагіна, призначеного для конструювання віртуальних лабораторних робіт.

3.3.4 Turbulenz

Бібліотеки двигуна реалізовані та оптимізовані на JavaScript для підтримки

швидкої інтеграції коду і даних. Двигун виконується безпосередньо в браузері, і включає в себе деякі з наступних функцій:

Асинхронне завантаження ресурсів і обмінювати; Лінива оцінка оновлень сцени; Багатопоточна перевірка і виконання.

Візуалізація ефектів і частинок. Shader на основі режиму негайного відправлення; Підтримка мульти-техніка, багатоходової, мульти-матеріали; Динамічна вершина, індекс і обробки текстур буфера; Відкладений надання підтримки необмежені вогні; Знімні ефекти POST і колекція ефектів; Експонентний карти тіней і оклюзія запитів; Великі частинок і ефект системи; GUI система / HUD підтримки декількох мов і шрифтів. Все це дає змогу створити дуже детальну візуалізацію певних процесів, що чудово в випадку створення ВЛ. Оскільки робота з UI при створенні проекту ВЛ є досить об'ємною і становить приблизно 30% від всього. Інтерфейс двигуна представлений на рис. 3.3.

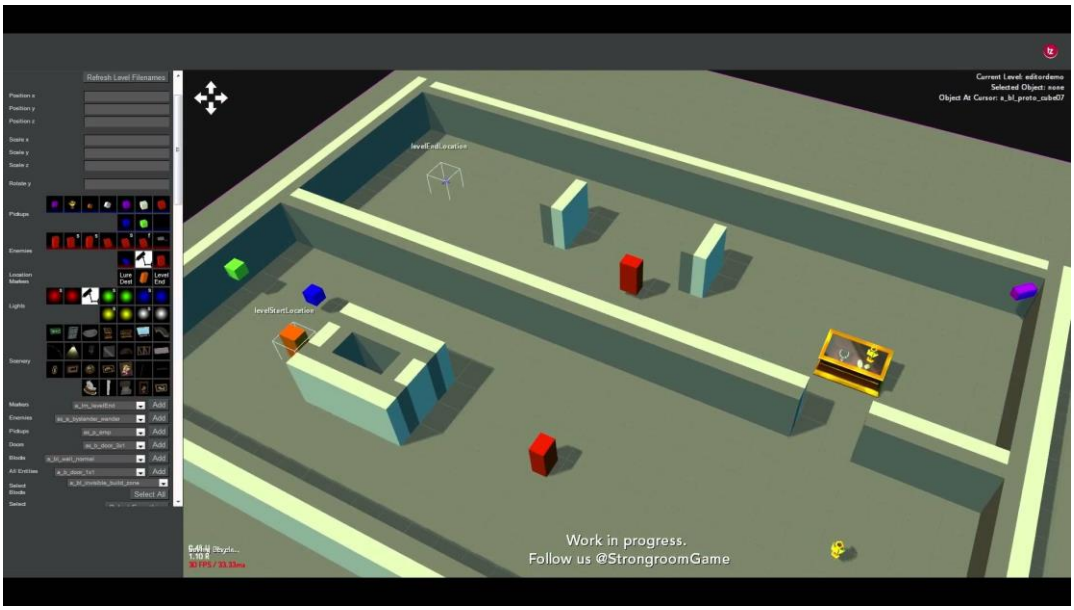

Рисунок 3.3 – Інтерфейс двигуна Turbulenz.

Фізика, Зіткнення і анімація. Жорсткі тіла, зіткнення примітиви і обмежень; Рей і опуклі запити розгортки; Велика колекція вбудованих

контролерів анімації; Скелет анімації з кватерніонів

Аудіо та Переферія. 3D джерел звуку і до 7.1 об'ємного звуку; багатопотокова передача і змішування; Доступ до HID, що дозволяє зовнішні контролери і периферійні пристрої

Мережа, Мультиплеер і соціальні мережі. Стиснення, шифрування, надійної і ненадійною передачі повідомлень; Компенсація мережу лаг, клієнт / сервер і p2p архітектури; Інтеграція з популярними соціальними мережами, включаючи Facebook; Автоматичний програвач входу і доступу автоматично обробляються.

Сцена і управління ресурсами. Видимість запитів через портали, усіченого або перекриттям коробки; Сортування і угрупування для видимості і оптимальної обробки; Пропускна здатність і апаратне масштабування за допомогою динамічного вибору активів[7].

3.3.5 Порівняльна характеристика

Тепер порівняємо усі аспекти кожного з ігрових рушіїв і подивимось, який доцільно обрати для реалізації тестового проекту, що зображено в табл.3.1.

Табиця 3.1 - Порівняння ігрових рушіїв

Критерії/ Альтернативи	VR	Зручність UI	Фізичний двигун	Мова програмування	Додаткові сервіси	Платформи	WWW сервіси
Unreal Engine	+(Google SDK)	3.5	+	C++, Blueprint system	AssetStore	Cross	+
Shiva 3D	Android VR	4	+	LUA	-	Cross	+
Unity	+(Cardboard SDK	5	+	C#/JS	UNET, CLOUD BUILD	Cross	UNET +.NET 3.5
Turbulenz	-	4	+	JS	-	Cross	+

3.4 Фреймворки

3.4.1 XNA

Microsoft XNA (англ. XNA's Not Acronymed) — набір інструментів з керованим середовищем часу виконання (.NET), створений Microsoft для полегшення розробки додатків. Мета XNA в спробі звільнити розробку від написання «повторюваного шаблонного коду» і об'єднати різні аспекти розробки в одній системі. Набір інструментів XNA був анонсований 24 березня 2004 на Game Developers Conference в Сан-Хосе, Каліфорнія. Перший Community Technology Preview XNA Build був випущений 14 березня 2006.

XNA Framework ґрунтується на реалізації. NET Compact Framework 2.0 для розробки для Xbox 360 і .NET Framework 2.0 на Windows. Він включає великий набір бібліотек класів, специфічних для розробки ігрових додатків, що підтримує максимальне повторне використання коду на всіх цільових платформах. Фреймворк виконується на модифікації Common Language Runtime, що оптимізована для ігор. CLR доступне дляWindows XP, Windows Vista, і Xbox 360. Так як проекти XNA пишуться для CLR, вони можуть бути запущені на будь-якій платформі, яка підтримує XNA Framework з мінімальними змінами або взагалі без них. Проекти, які запускаються на фреймворку, технічно можуть бути написані будь-якою .NET-сумісною мовою, але офіційно підтримується тільки мова програмування C# та середовище швидкої розробки XNA Game Studio Express і всі версії Visual Studio 2008.

XNA Framework приховує низькорівневі технологічні деталі, пов'язані з розробкою. Таким чином, фреймворк піклується про різницю між платформами, дозволяючи розробникам приділяти більше уваги смисловому вмісту додатку. XNA Framework інтегрується з декількома інструментами, такими як XACT, для допомоги в створенні контенту. XNA Framework надає підтримку створення та двомірних, і тривимірних ігор і дозволяє використовувати можливості

контролерів Xbox 360. Десктопні програми можуть поширюватися безкоштовно під поточним ліцензуванням Microsoft.

3.4.1 JAVA FX

JavaFX створена на базі технології Java. JavaFX розширює можливості Java, дозволяючи розробникам використовувати будь-яку бібліотеку Java в JavaFX-додатках.

Дана технологія дозволяє користувачам бачити JavaFX-додатки в веб-браузері або взагалі не використовувати веб-оглядач, перетягнувши такий додаток на робочий стіл.

Вона забезпечує ефективну взаємодію між дизайнерами та розробниками за допомогою утиліти Project Nile: дизайнери можуть працювати зі своїми звичайними інструментами і при цьому взаємодіяти з творцями веб-сценаріїв, які використовують середу NetBeans IDE разом з JavaFX;

Платформа дозволяє розробникам створювати насичені інтерактивні додатки для різного інформаційного наповнення, насичені векторною графікою, анімацією, аудіо та відео.

3.6 Висновки

Ігрові двигуни безсумнівно виграють в зручності над такими технологіями, як чистий веб-JS та фреймворками. Оскільки їхні IDE інтуїтивно зрозумілі і всі необхідні компоненти і сервіси не потрібно інсталювати і налаштовувати під себе з нуля. Встановивши ігровий рушій ви отримуєте все і одразу, а додаткові сервіси лише спростять задачу розробки ВЛ.

За основу реалізації тестового проекту було обрано Unity3D та Google Cardboard SDK. Кросплатформеність, готові реалізації функцій та Google API для VR. Можливість кастомізовувати Editor, Unity Cloud Build та UNET виводять Unity3D в лідери із списку.

4 UNITY ТА VR В СИСТЕМІ ДИСТАНЦІЙНОГО НАВЧАННЯ

Створення тестового проекту з використанням Unity3D та Google Cardboard SDK і вбудованих сервісів.

4.1 Віртуальна реальність

Віртуальна реальність (Virtual reality) — уявна реальність, створена за допомогою комп'ютерних систем, які забезпечують візуальні і звукові ефекти, що занурюють глядача в ілюзорний світ за екраном. Користувач оточується породженими комп'ютером образами і звуками, що дають відчуття реальності. Користувач взаємодіє зі штучним світом за допомогою різноманітних сенсорів, таких як, наприклад, шолом і рукавички, які зв'язують його рухи, враження і аудіовізуальні ефекти. Майбутні дослідження в галузі віртуальної реальності скеровані на збільшення враження реальності спостережуваного.

4.1.2 Застосування

Звичайно найпопулярнішою нішою VR є відеоігри, але не тільки забави є сферою застосування віртуальної реальності в наш час. Завдяки операціям на віртуальних пацієнтах лікарі вивчають нові методики і хірургічні техніки. Віртуальні тренажери допомагають пілотам літаків або водіям відпрацьовувати ситуації, потрапити в які практично неможливо в сучасному світі навіть людині з великим досвідом. Віртуальне відвідування музеїв стане скоро таким же звичним, як і 3D-кінотеатри. Завдяки 3d окулярам віртуальної реальності ми зможемо побачити Собор Паризької Богоматері або музей Пергамон в Берліні, подивитися на римський Колізей або близько підійти до Джоконди, до якої, до слова, в головному музеї Парижа проштовхатись через натовп дуже проблематично.

Рисунок 4.1 – Демонстрація віртуальної реальності [4]

Вже сьогодні існує маса навчальних відео і програм у VR (рис. 4.1). Проте віртуальна реальність може не лише давати знання, але й навчити тому, як робити не слід. Operation Lifesaver (OL) – проект, який наочно пояснює, чому не варто переходити залізницю поза спеціально обладнаними місцями або порушуючи застережливі знаки. Подивитися ролик можна як в VR, так і просто на моніторі.

Нью-йоркська компанія GeoCV отримала $1,8 млн інвестицій. Суть проекту – 3D-сканування приміщень для подальшого використання зображення в режимі VR дизайнерами, будівельниками, архітекторами і так далі.

VR First і Crytek навіть вирішили створити 50 VR- лабораторій по всьому світу до кінця 2017. Мета лабораторій – розвиток віртуальної освіти, стимулювання ринку праці AR і VR, створення нового і якісного контенту. Лабораторії обладнають найновішим устаткуванням. Одна з лабораторій знаходитиметься в Києві.

4.1.3 Google VR SDK for Unity

Тісна інтеграція з Unity Google VR дозволяє легко створювати Android додатки для Daydream і Cardboard. Google VR SDK для Unity надає додаткові

функції, такі як спеціалозоване аудіо(VR), підтримка контролера Daydream, різноманітні сервіси та тестові приклади.

Нативна підтримка Unity для Google VR дозволяє легко:

- Почати ВР проект з нуля.
- Адаптувати існуючий Юніті проект під ВР
- Зробити додаток, який може легко перемикатися в і з режиму VR
- Інтеграція з Google VR забезпечує:
- Відстеження голови
- Стереорендерінг
- Виявлення взаємодії користувача з системою (через тригер або контролер).
- Автоматична настройка стерео для конкретного глядача VR.
- Корекція спотворень для лінз глядача в VR.
- Автоматична корекція дрейфу гіроскопів.

Google VR SDK для Unity надає наступні додаткові можливості:

- Емуляція VR в режимі відтворення редактора Unity, використовуючи мишку і альт / клавіші управління для панорами або нахилу камери.

Дисплей FPS показує продуктивність рендеринга вашого девайсу[8].

4.2 Задачі тестового проекту

Окреслимо основні задачі тестового проекту для більш детального аналізу та демонстрації обраного інструментального засобу Unity3D.

Перш за все проект має показати можливості роботи з VR, можливості взаємодії додатку із сервером та базою даних, а також фізичного рушія. При аналізі слід приділити увагу співвідношенню отриманого результату зі складністю його імплементації.

По-друге ми маємо продемонструвати роботу з вбудованими можливостями Unity3D: WEBGL build, Cloud build.

По-третє потрібно освітлити роботу з логікою побудови додатків в системі

ДН, взаємодію зі сценою, об‘єктами та інтерфейсом.

4.3 Архітектура тестового проекту

Загальний вигляд клієнт-серверної архітектури ВЛ показано на рис.

Сервер та клієнт обмінюються даними за допомогою REST запитів(клас WWW unity). Оскільки БД знаходиться на сервері за збереження даних буде відбовідати сервер і у разі необхідності надавати дані користувачу в залежності від його дій, що зображено на рис. 4.1.

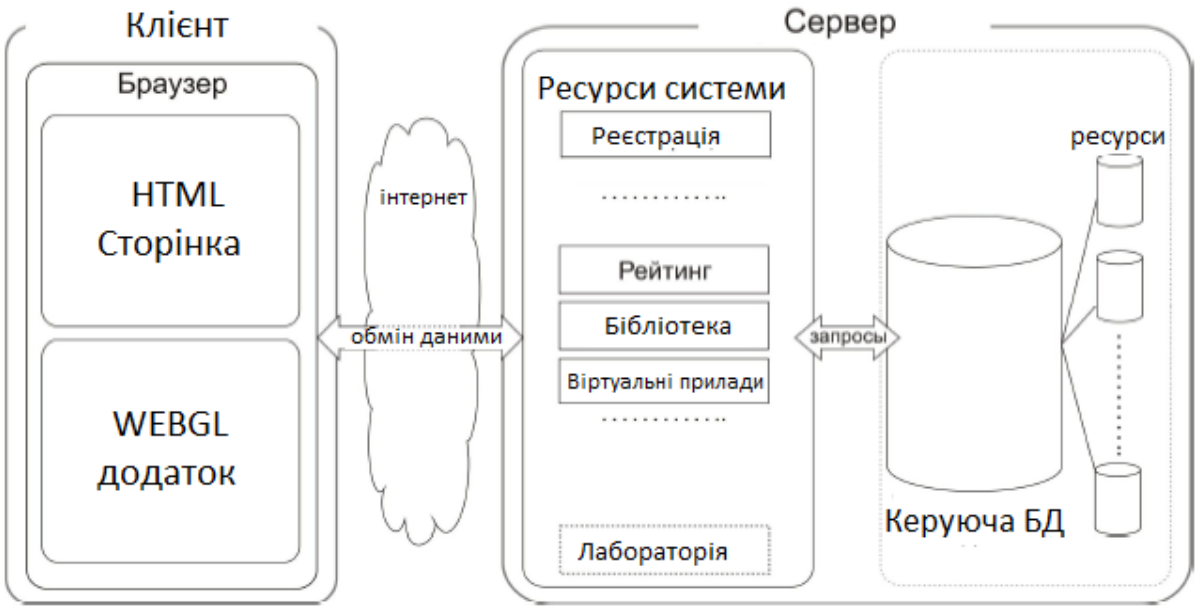

Рисунок 4.1 - Загальний вигляд архітектури ВЛ.

Віртуальна лабораторія містить в собі значний обсяг інформації, яка при першому підході може вимагати багато дискового простору і оперативної пам'яті, що безумовно буде займати зайві ресурси при кожному запуску програми.

Користувачеві часто всі модулі відразу не потрібні, в цьому випадку оптимальним варіантом буде закласти в системі можливість завантаження тільки основної оболонки, визначеного робочого модуля і необхідного розділу

бібліотеки об'єктів. Тоді користувач заощадить трафік ресурсів і зможе підвищити ефективність роботи.

Представлена на рис. 4.2 схема являє собою архітектуру програмного модуля «Віртуальна лабораторія». У серверної частини програми доцільно розглядати три основних компонента - ядро модуля, підключення бази даних і математичний процесор.

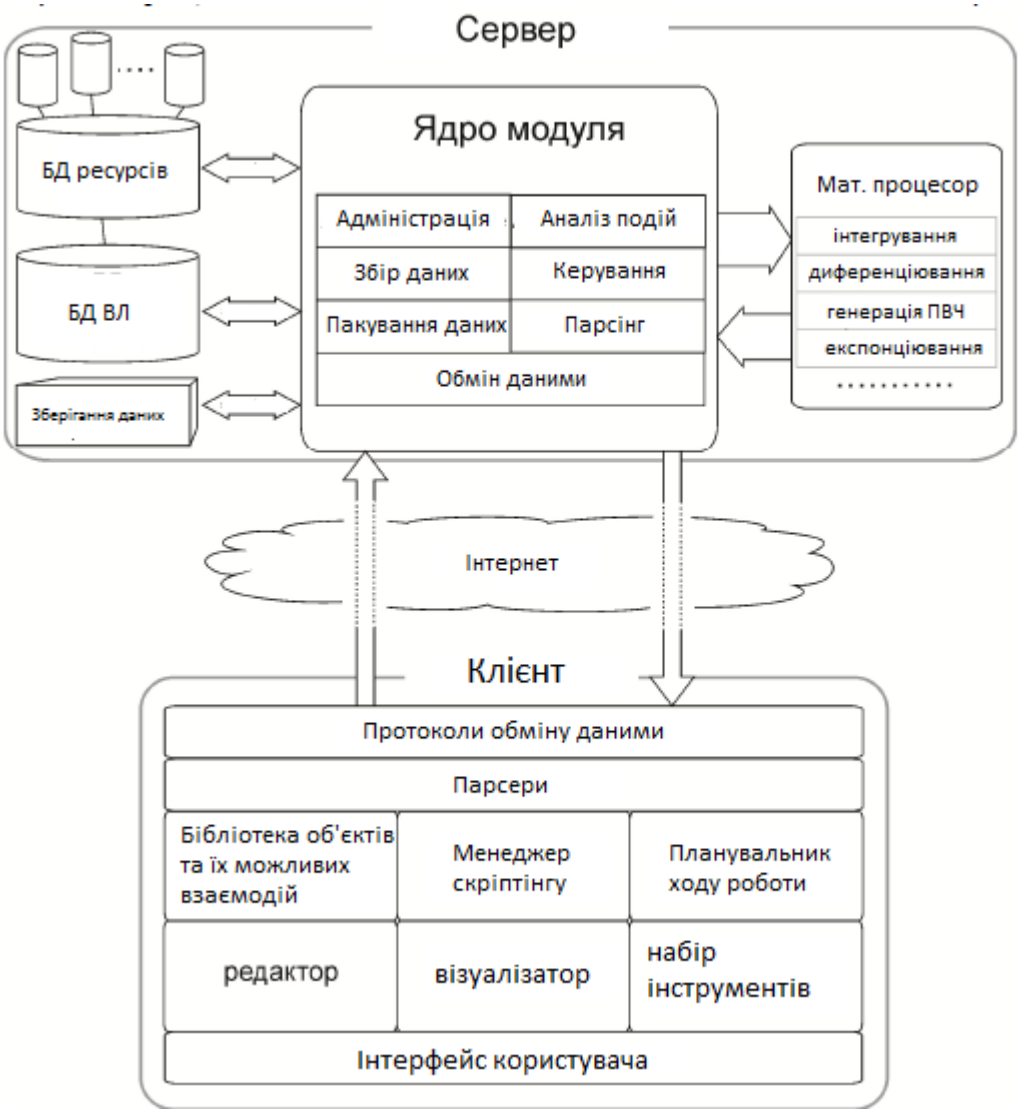

Рисунок 4.2 - Детальна архітектурна схема.

В ядрі модуля міститься набір основних класів управління:

1. «Адміністрування» відповідає за систему безпеки, права користувачів,

управління обліковими записами користувачів, забезпечує синхронізацію їх ресурсів, підключає налаштування компонентів, що використовуються конкретним користувачем.

2. «Аналізатор подій» записує, впорядковує і зберігає всі дії користувачів для подальшого відтворення.

3. «Збір даних» на початку роботи забезпечує створення пакетів даних і файлів ресурсів для бібліотеки користувача з подальшою відправкою в репозитарій сервера.

4. «Керування» створює віртуальні кімнати, підключає до них користувачів, забезпечує авторизацію і адміністрування кімнат. Відповідає за прийом
і передачу даних між користувачами і сервером.

5. Модуль «пакування даних» створює пакети з ресурсів, які будуть відправлені
клієнту, а також для забезпечення цілісності даних створює метадані цих пакетів ресурсів.

6. Модуль «парсінг» забезпечує перетворення запитів, що надходять і повідомлень від клієнтського додатка і передачу цих даних керуючим модулів.

7. «Обмін даними» - це інтерфейс веб-сервісів взаємодії з клієнтськими додатками.

БД системи

Перш за все, це база авторизації, статистики, також здійснюється Передавання даних про прогрес користувачів в базу оцінювання і т.п. Крім використовуваних баз СДН, модулю необхідна власна база даних. У ній зберігаються записи про роботу всіх компонентів модуля, інформація про об'єкти, записи подій, а також адреси зв'язку з сховищами ресурсів бібліотек. Математичний процесор служить для забезпечення складних математичних розрахунків, які не можуть виконуватися на клієнтських машинах. Це розрахунки

рішення нелінійних рівнянь, розрахунки, що вимагають використання математичних методів кореляції, уточнення, ітерацій, підрахунків похибок, завдання аналізу, і інші[9].

Клієнтська частина програмного забезпечення ВЛ містить в собі компоненти для створення і роботи з віртуальними моделями:

1. «Бібліотека об'єктів і взаємодій» містить набори об'єктів певної предметної області, а також способи взаємодії, які визначаються даної предметної областю.

2. «Менеджер скріптінгу» дозволяє налагоджувати взаємодії між об'єктами, задавати їх параметри, накладати умови. Також цей компонент призначений для опису властивостей і взаємодій нових об'єктів, що створюються в редакторі.

3. В «редакторі» ВЛ виробляється конструювання нових об'єктів. В ньому користувачі можуть на основі існуючих об'єктів бібліотеки, створювати свої об'єкти. Користуючись паралельно компонентом скріптінгу, користувачі можуть уточнювати і дописувати нові параметри і методи взаємодії з іншими об'єктами, дописувати нові способи поведінки.

4. «Візуалізатор» компонент, в якому відбувається лабораторне дослідження, забезпечує відображення всіх конструкцій об'єктів. У цьому модулі також відображаються поточні динамічні дані, такі як значення параметрів системи, вказуються результуючі напрямки рухів, і ін. Безпосередньо тут відбувається сам процес моделювання та управління ходом роботи.

5. «Набір інструментів» - це призначені для користувача засоби управління і зміни об'єктів, що знаходяться в компонентах Візуалізатор і Редактор.

6. «Інтерфейс користувача» - відповідно сукупність методів відображення і взаємодії користувача і системи.

4.2.1 Загальний підхід до реалізації

Сучасні можливості мережевих технологій, повсюдне поширення високопропускними телекомунікаційних каналів зв'язку, а також наявність широких можливостей засобів програмування, в яких стало комфортно розробляти веб-орієнтовані програмні продукти, дозволяють говорити про те, що віртуальні навчальні засоби сьогодні доцільно створювати у вигляді веб-сервісів. З огляду на той факт що СДН є Інтернет додатком, при розробці ВЛ буде очевидна така ж схема реалізації - під контейнер браузера. Хоча, не слід відкидати можливість створення програмного забезпечення адаптованого під локальну мережу з можливістю при роботі зв'язуватися з сервером для обміну даними. Схема реалізації представлена на рис. 4.3.

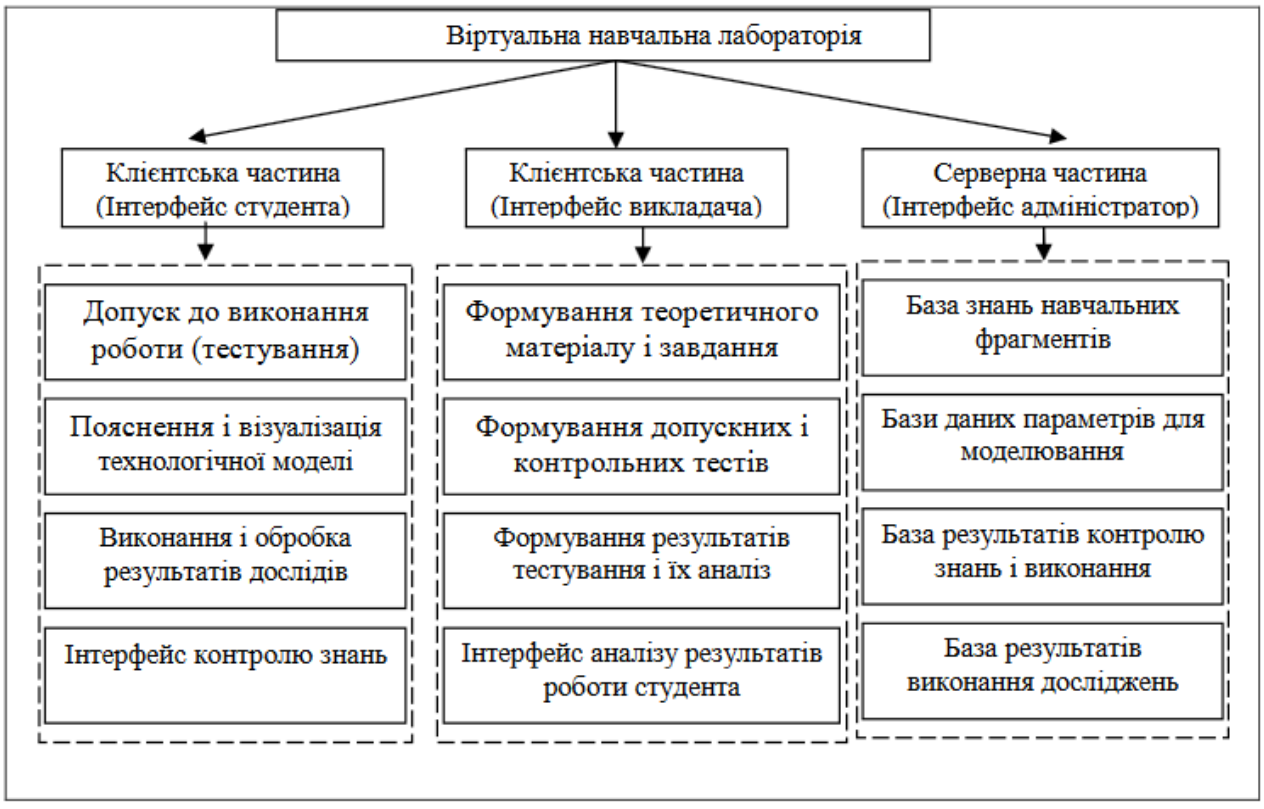

Рисунок 4.3 – Схема реалізації ВЛ.

4.2.2 Реалізація

Процес розробки тестової віртуальної лабораторної роботи засобами редактора Unity3D, буде складатися з наступних кроків:

- додавання об'єкта в 3-мірну сцену;

- визначення унікального імені для об'єкта, за допомогою якого користувач буде звертатися до об'єкта;

- визначення переліку параметрів, необхідних фізичного об'єкту для участі в експерименті;

- визначення додаткових змінних або констант (прискорення вільного падіння, нормальна або поточна температура середовища та інше);

- написання математичних формул для моделювання процесу або явища;

Функціональність віртуальних лабораторних робіт. Кожна лабораторна робота містить окремі фізичні об'єкти, кожен фізичний об'єкт має власну математичну модель, тому користувач повинен мати можливість змінювати математичну модель. На рис. 4.4 представлена робоча область ВЛ.

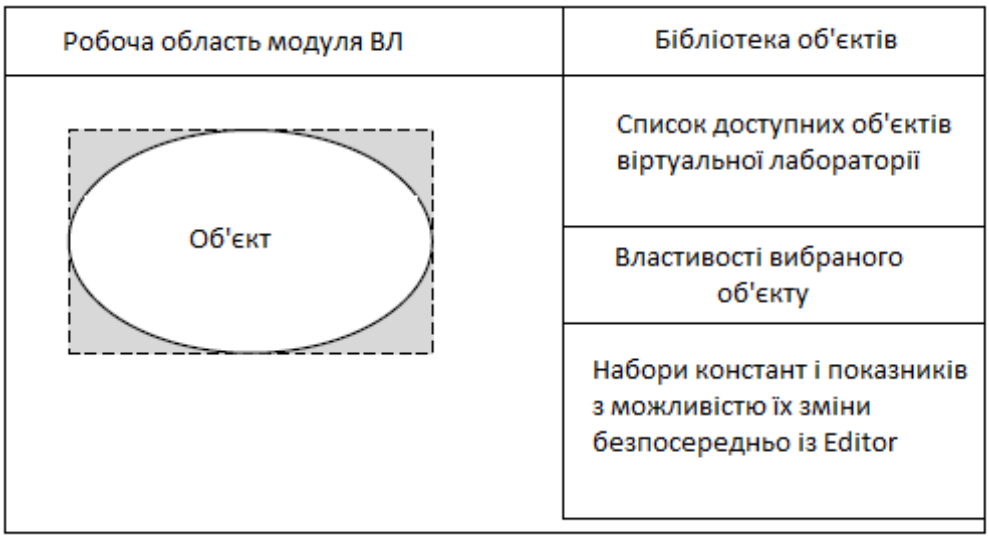

Рисунок 4.4 – Робоча область ВЛ.

При створенні або редагуванні ВЛР користувач виділяє об'єкт зі списку об'єктів на сцені. На рис. 4.5 представлена загальна схема ВЛ.

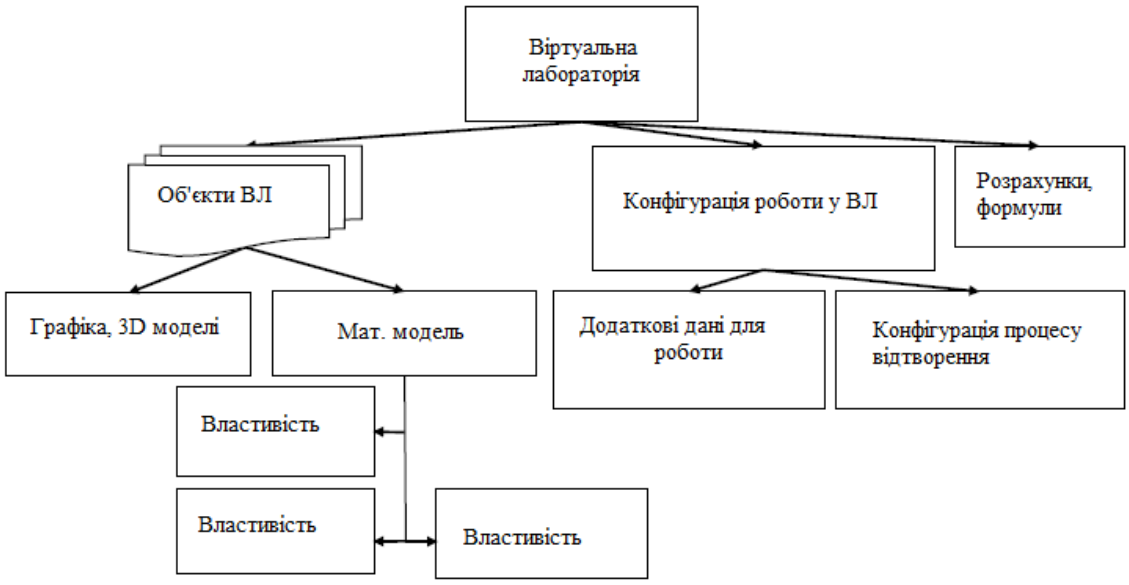

Рисунок 4.5 – Загальна схема ВЛ.

Після цього у вікні властивостей об'єкта виводиться список властивостей поточного об'єкта, і відображаються елементи інтерфейсу, за допомогою яких користувач може змінити значення даних властивостей.

Додавання пакету GoogleVR виконуєтья простим імпортом пакетів рис. 4.6.

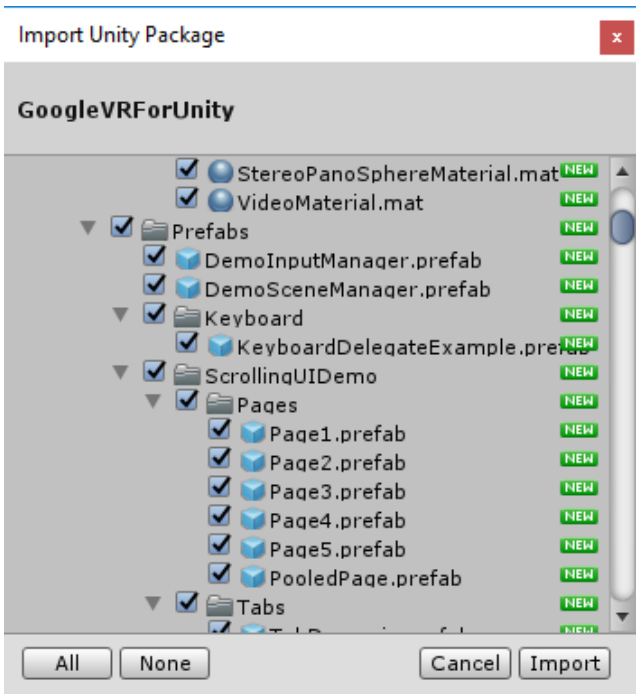

Рисунок 4.6 – Імпорт пакету Google VR.

Далі ми готові налаштовувати View для нашого користувача Cardboard.

Об’єкт, що відповідає за імітацію роботи ВР пристрою називається **GvrViewerMain.**

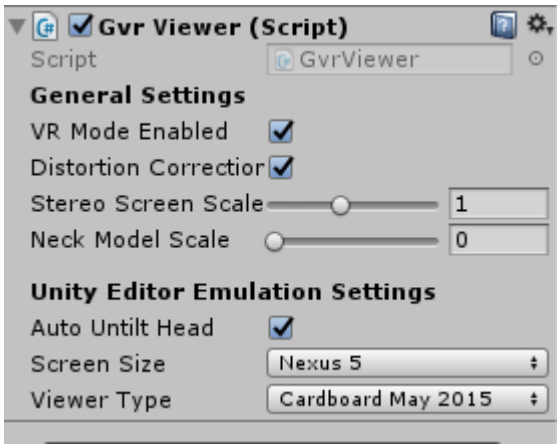

Рисунок 4.7 – Налаштування GvrViewerMain.

Як видно з рис. 4.7 Ми можемо обирати прилад емуляції в Юніті а також тип та версію Cardboard.

Запустивши проект на рис. 4.8 ми бачимо емуляцію пристрою ВР на ПК безпосередньо у Unity Editor

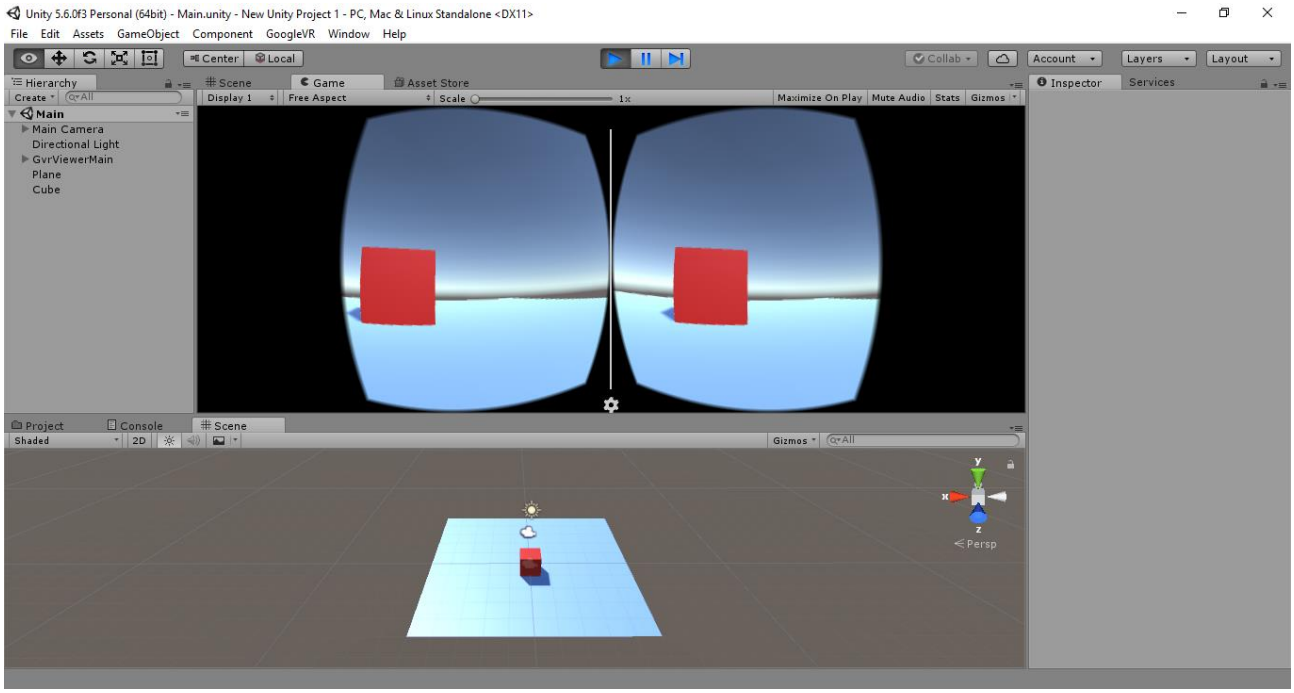

Рисунок 4.8 – Запуск проекту, емуляція приладу ВР.

В Unity є декілька вбудованих рушіїв для 2D та 3D та декілька способів взаємодії з ними. Перш за все потрібно додати компонент Capsule Collider до об‘єкту гри, тип Collider буде визначати тип рушія, для прототипу використаємо Physics Collider на об’єкт користувача кардбоард рис. 4.9. Цей колайдер буде використовуватись для визначення претину та доторкання об‘єктів з іншими колайдерами, додаткові налаштування triggers дозволяє обмету проходити через інші об‘єкти проте так само викликати подію накладіння, напроти Cinematic objects буде зупиняти об‘єкти що доторкаються до нього, проте не будуть викликати подію накладання якщо будуть рухатись самі.

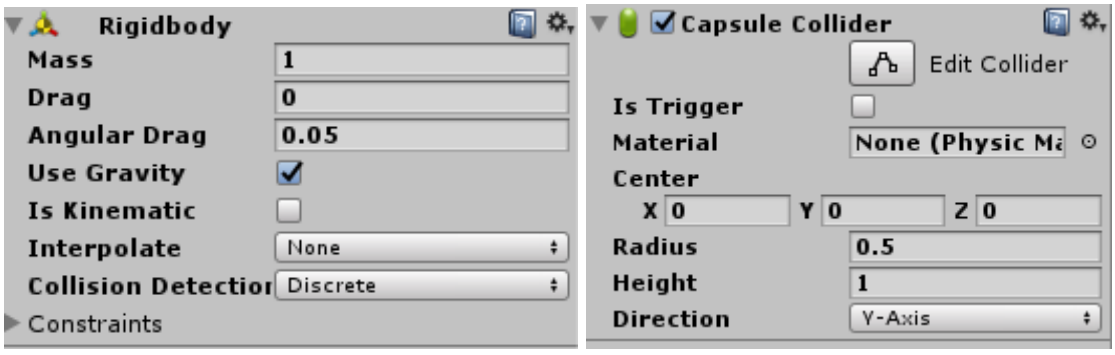

Рисунок 4.9 – Налаштування фізики для об’єктів.

Налаштування для створеного маятника зі скріпта представлені на рис. 4.10.

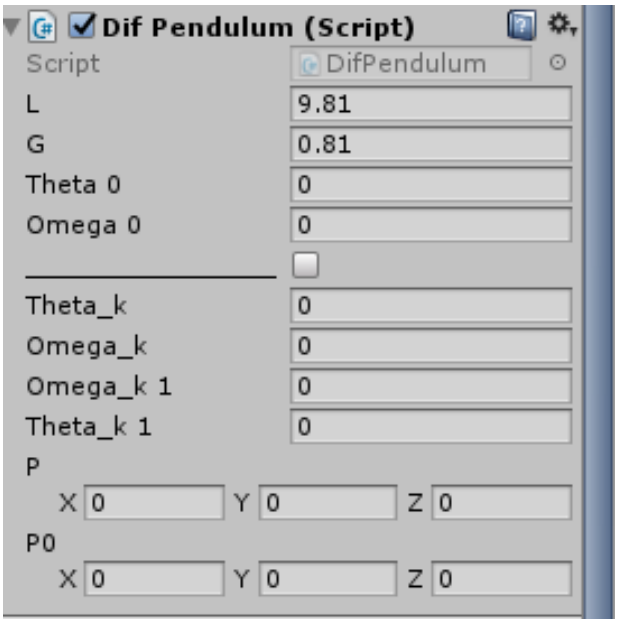

Рисунок 4.10 - Налаштування користувача ВЛ для маятника

Реалізація математичної моделі маятника представлено на рис. 4.11.

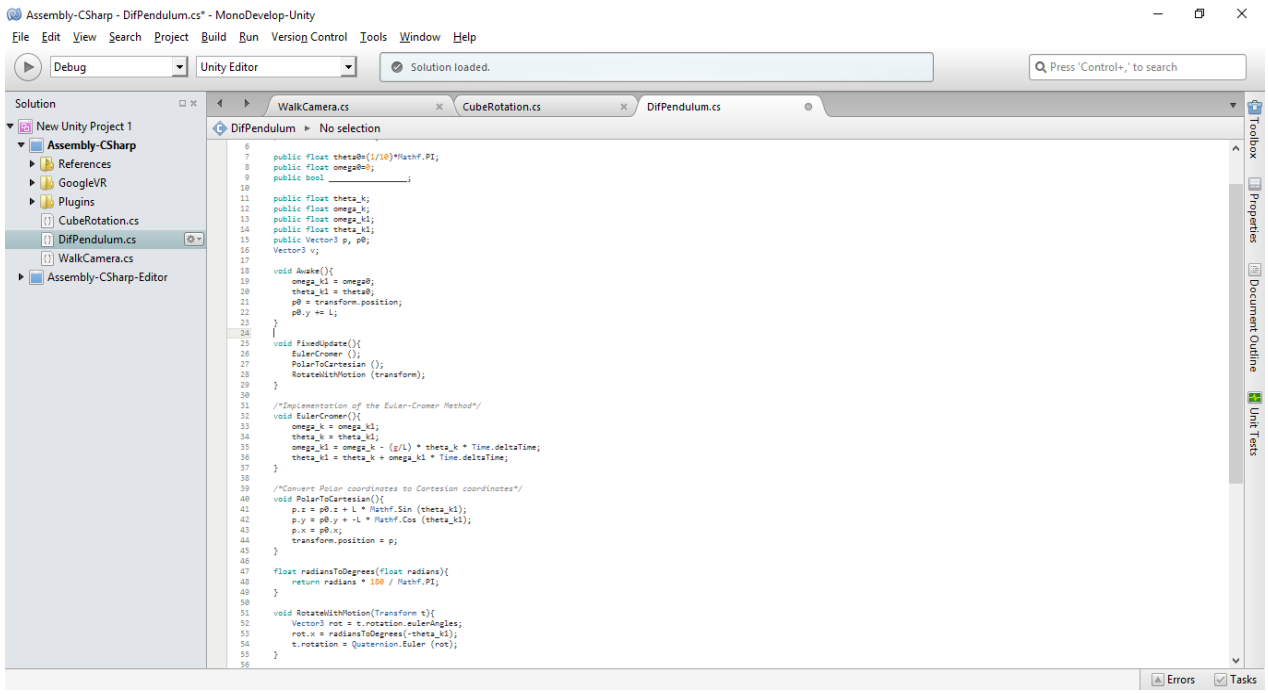

Рисунок 4.11 - Реалізація мат. моделі на C#.

Реалізація керування віртуальним об’єктом за допомогою клавіш на пристрої ВР представлена на рис. 4.12.

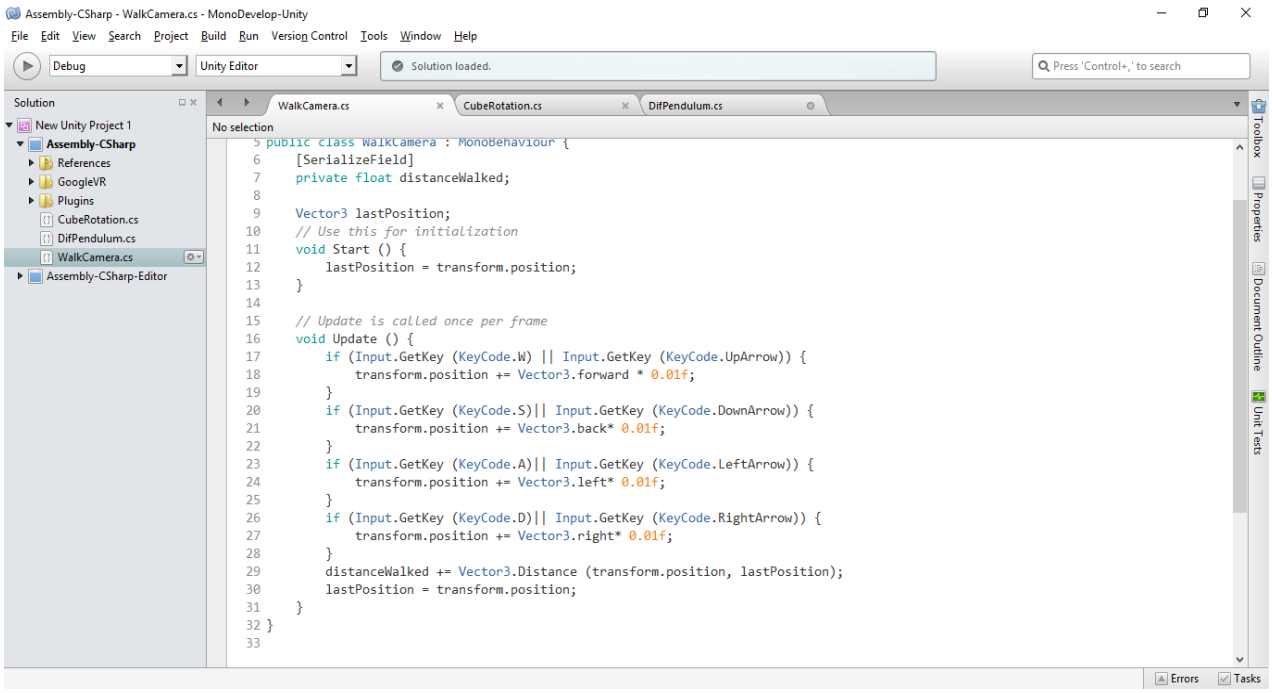

Рисунок 4.12 - Керування користувача VR.

Реалізація керування об’єктом з допомогою клавіш на пристрої Cardboard.

Приклад створення і відправки простого запиту за допомогою C# і класу WWW в Unity представлено на рис. 4.13.

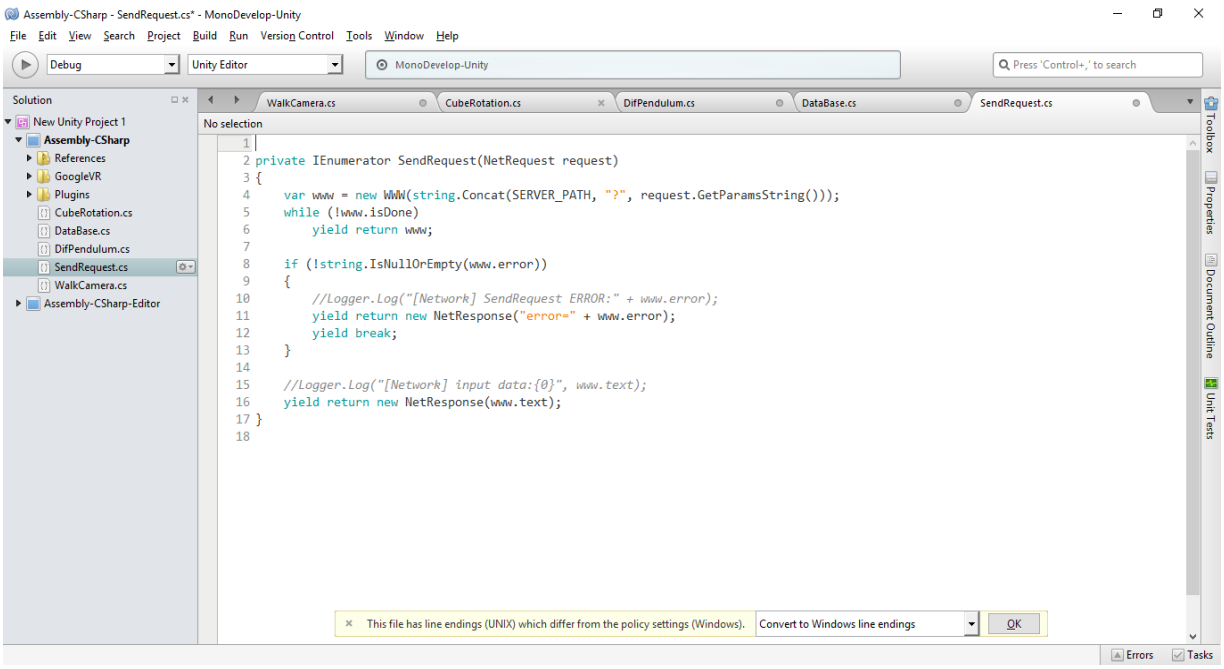

Рисунок 4.13 – Запит з класу WWW.

Було використано бд - sql lite. Відбувається запис пройденої дистанції кожним користувачем окремо на рис. 4.14.

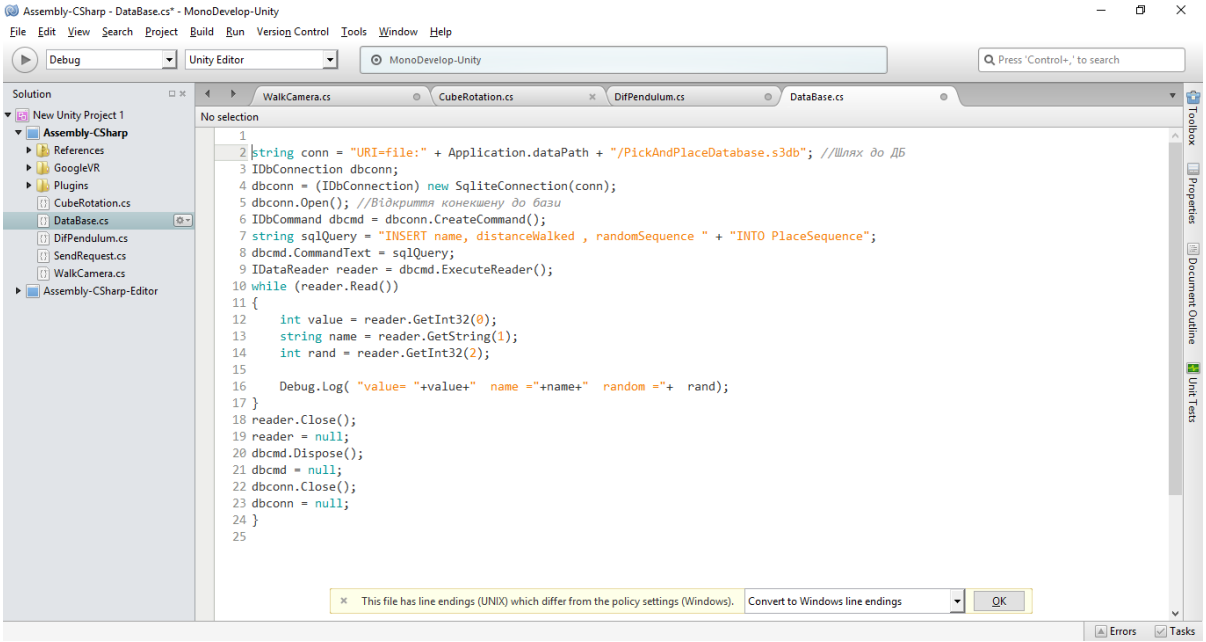

Рисунок 4.14 – Робота з БД.

4.3 Відображення тестового проекту

Можливість зібрати проект під WebGL дає змогу розмістити додаток на сервері показано на рис.4.15.

Рисунок 4.15 - Білд WebGL.

Модуль для відображення лабораторних робіт є веб-сторінку, яка виконує роль контейнера для інтерактивного мультимедійного об'єкта віртуальної лабораторії. На цій HTML-сторінці представлений вбудований програмний об'єкт, функціональність якого забезпечується тим, що підключається плагіном Unity-Webplayer (рис. 4.16).

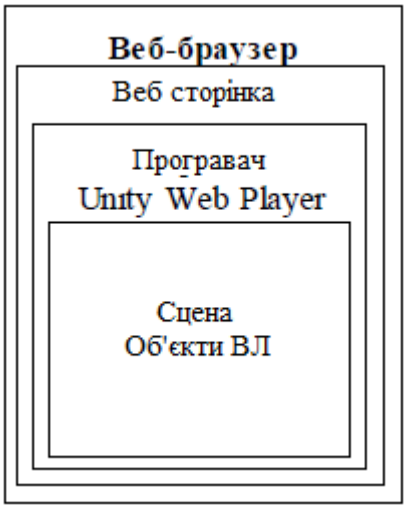

Рисунок 4.16- Відображення контенту

Для проведення математичних розрахунків інтерактивний об'єкт взаємодіє з математичним пакетом, який може функціонувати тільки в рамках веб-сервісу, тому сторінка-контейнер повинна мати необхідний програмний інтерфейс, що допоможе в передачі вхідних даних для розрахунків в математичний пакет, а після розрахунків поверне результати в інтерактивний об'єкт для відображення нових даних.

4.4 Висновки

Можливості .Net в поєднанні з Unity Editor мають дуже високий потенціал і бузсумнівно є одним з найкращих інструментів для створення проекту віртуальної лабораторії. Легкість налаштування VR проекту(можливість впроавдження до існуючого). Зручна система колайдерів та фізичний двигун ідеально підходять для симуляції фізичних явищ та процесів. Можливість зробити колайдер по моделі 3D об’єкта дає змогу симулювати фізичні процеси

для будь яких за складністю об’єктів. Можливість контролю фізики за допомогою написання скріптів та створення програмних мат. Моделей для об’єктів доводить, шо використиння ігрових рушіїв зокрема Юніті ідеально підходить для створення проектів ВЛ. Також слід не забувати про ком’юніті та форуми unity3D. Там завжди можна знайти відпоіді на найважчі вопроси та гайди по використанню тих-чи інший плагінів та розширень для двигуна. Можливості C# дозволяють реалізовувати клієнт-серверну взаємодію. Оскільки Unity повністью підтримує .NET3.5 робота з базами даних і відправкою запитів до сервера через наприклад через вбудований клас WWW.

5 ВАРТІСНИЙ АНАЛІЗ ВИКОРИСТАННЯ ТЕХНОЛОГІЙ

У даному розділі проводиться оцінка основних характеристик програмного продукту, призначеного для демонстрації можливостей створення ВЛ за допомогою ігрового рушія Unity3D.

Програмний продукт призначено для використання на персональних комп’ютерах під управлінням будь-якої операційної системи.

Функціонально-вартісний аналіз (ФВА) – це технологія, яка дозволяє оцінити реальну вартість продукту або послуги незалежно від організаційної структури компанії. Як прямі, так і побічні витрати розподіляються по продуктам та послугам у залежності від потрібних на кожному етапі виробництва обсягів ресурсів. Виконані на цих етапах дії у контексті метода ФВА називаються функціями.

Мета ФВА полягає у забезпеченні правильного розподілу ресурсів, виділених на виробництво продукції або надання послуг, на прямі та непрямі витрати. У даному випадку – аналізу функцій програмного продукту й виявлення усіх витрат на реалізацію цих функцій.

Фактично цей метод працює за таким алгоритмом: визначається послідовність функцій, необхідних для виробництва продукту. Спочатку – всі можливі, потім вони розподіляються по двом групам: ті, що впливають на вартість продукту і ті, що не впливають. На цьому ж етапі оптимізується сама послідовність скороченням кроків, що не впливають на цінність і відповідно витрат.

- для кожної функції визначаються повні річні витрати й кількість робочих часів.
- для кожної функції наоснові оцінок попереднього пункту визначається кількісна характеристика джерел витрат.

- після того, як для кожної функції будуть визначені їх джерела витрат,

проводиться кінцевий розрахунок витрат на виробництво продукту.

5.1 Постановка задачі техніко-економічного аналізу

У роботі застосовується метод ФВА для проведення техніко-економічний аналізу розробки системи аналізу нелінійних нестаціонарних процесів. Оскільки основні проектні рішення стосуються всієї системи, кожна окрема підсистема має їм задовольняти. Тому фактичний аналіз представляє собою аналіз функцій програмного продукту, призначеного для збору, обробки та проведення аналізу гетероскедастичних процесів в економіці та фінансах.

Відповідно цьому варто обирати і систему показників якості програмного продукту.

Технічні вимоги до продукту наступні:

- програмний продукт повинен функціонувати на персональних

комп‘ютерах із стандартним набором компонент;

- забезпечувати високу швидкість обробки даних та відклик користувачеві у

реальному часі;

- забезпечувати зручність і простоту взаємодії з користувачем або з

розробником програмного забезпечення у випадку використовування його як модуля;

- передбачати мінімальні витрати на впровадження програмного продукту.

5.2 Обґрунтування функцій програмного продукту

Головна функція F_0– розробка програмного продукту, який аналізує процес за вхідними даними та будує його модель для подальшого прогнозування. Виходячи з конкретної мети, можна виділити наступні основні функції ПП:

F_1 – вибір мови програмування;

F_2 – вибір ігрового рушія;

F_3 – вибір тематики ВЛ

Кожна з основних функцій може мати декілька варіантів реалізації.

Функція F_1:

а) JS;

б) C#;

Функція F_2:

а) Shiva3D

б) XNA

в) Unity

Функція F_3:

а) Фізика;

б) Природні явища;

в) Машинобудування;

5.3 Варіанти реалізації основних функцій

Варіанти реалізації основних функцій наведені у морфологічній карті системи (рис. 5.1). На основі цієї карти побудовано позитивно-негативну матрицю варіантів основних функцій (таблиця 5.1).

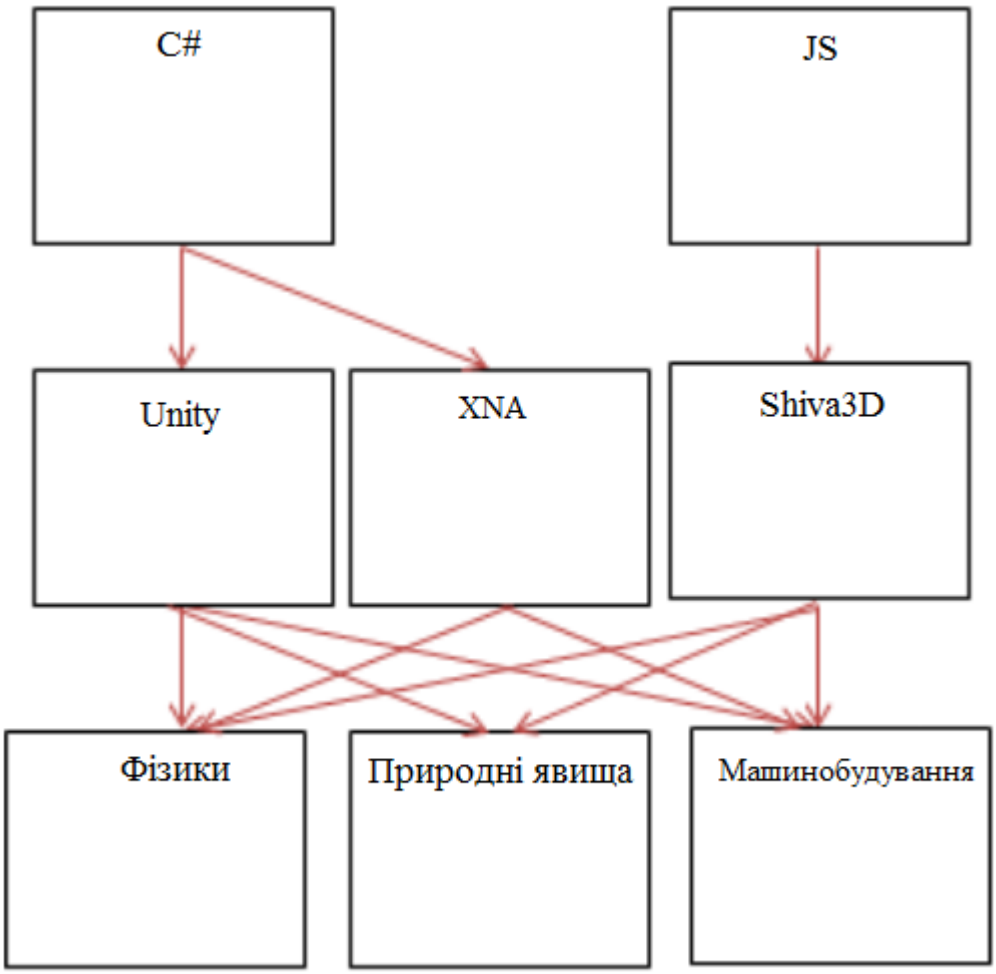

Рисунок 5.1 Морфологічна карта

На основі аналізу позитивно-негативної матриці робимо висновок, що при розробці програмного продукту деякі варіанти реалізації функцій варто відкинути, тому, що вони не відповідають поставленим перед програмним продуктом задачам. Ці варіанти відзначені у морфологічній карті.

Таблиця 5.1 – позитивно-негативна матриця варіантів основних функцій

Основні функції	Варіанти реалізації	Переваги	Недоліки
F1	А	Зручніше під час написання коду	Прив’язка до .Net
	Б	Низький поріг входу для розробника, кросплатформеність	Складніша відладка
F2	А	Потужний граф. та фіз. Двигун, зручна відладка, конвертація коду в С	Недостатня кількість бібліотек
	Б	Гарна документація, комюніті	Потребує більше самописного коду, мало готових рішень
	В	Вбудована підтримка багатьох платформ, багато бібліотек	Менше контролю над рендерінгом
F3	А	Вбудований фіз двигун, простота налаштування	Складні алгоритми обробки подій
	Б	Видовищність, цікавість	Складне моделювання процесів
	В	Широке практичне застосування, візуалізація	Потребує специфічних знань в області

Функція *F1:*

Оскільки в межах дипломної роботи ставиться демонстрація функціональних можливостей рушіїв , тому різноплановий підхід до написання тестового проекту ВЛ та використання декількох мов буде плюсом. Обираємо обидва варіанти А та Б .

Функція *F2:*

Написання прототипів за допомогою рішення Б займе надто багато часу та зусиль тому, відкинемо його.

Обираємо варіанти А та В .

Функція *F3:*

Згідно намічених задач, ми маємо продемонструвати можливості рушіїв з допомогою демо програм, тому варіант Б відкидаємо як недостатньо показовий, а варінт В як надто складний у реалізації для демо прототипу.

Таким чином, будемо розглядати такі варіанти реалізації ПП:

1. F1а – F2а – F3а

2. F1б – F2в – F3а

Для оцінювання якості розглянутих функцій обрана система параметрів, описана нижче.

5.4 Обґрунтування системи параметрів ПП

5.4.1 Опис параметрів

На підставі даних про основні функції, що повинен реалізувати програмний продукт, вимог до нього, визначаються основні параметри виробу, що будуть використані для розрахунку коефіцієнта технічного рівня.

Для того, щоб охарактеризувати програмний продукт, будемо використовувати наступні параметри: На підставі даних про основні функції, що повинен реалізувати програмний продукт, вимог до нього, визначаються основні

параметри виробу, що будуть використані для розрахунку коефіцієнта технічного рівня.

Для того, щоб охарактеризувати програмний продукт, будемо використовувати наступні параметри:

– *X1* – швидкодія мови програмування;

– *X2* – об‘єм пам‘яті для збереження даних;

– *X3* – час обробки даних;

– *X4* – потенційний об‘єм програмного коду.

X1: Відображає швидкодію операцій залежно від обраної мови програмування.

X2: Відображає об‘єм пам‘яті в оперативній пам‘яті персонального комп‘ютера, необхідний для збереження та обробки даних під час виконання програми.

X3: Показує розмір програмного коду який необхідно створити безпосередньо розробнику.

X4: Відображає час, який витрачається на дії.

5.4.2 Кількісна оцінка параметрів

Гірші, середні і кращі значення параметрів вибираються на основі вимог замовника й умов, що характеризують експлуатацію ПП як показано у табл. 4.2.

Таблиця 5.2 – Основні параметри ПП

Назва Параметра	Умовні позначення	Одиниці виміру	Значення параметра		
			гірші	середні	кращі
Швидкодія мови програмування	X1	Оп/мс	19000	11000	2000
Об’єм пам’яті для збереження даних	X2	Мб	32	16	8
Потенційний об’єм програмного коду	X3	Кількість строк коду	2000	1500	1000
Час обробки запитів користувача	X4	Мс	200	100	50

За даними таблиці 5.2 будуються графічні характеристики параметрів – рис. 5.2 – рис. 5.4.

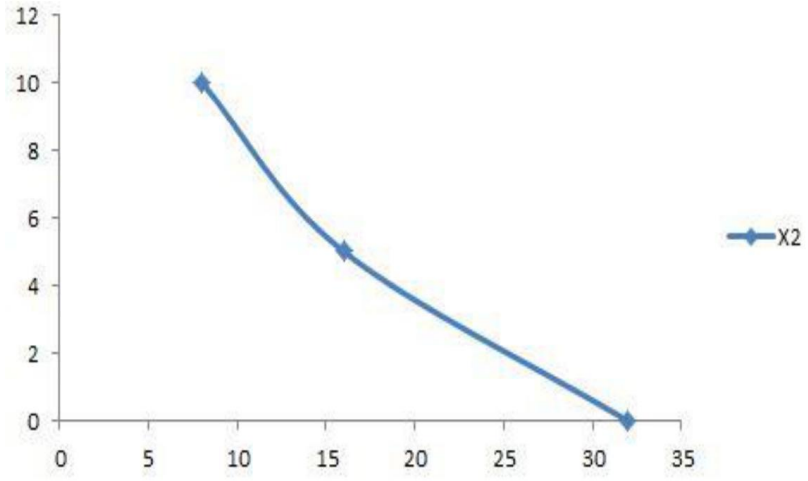

Рисунок 5.2 – X2, об‘єм пам‘яті для збереження даних

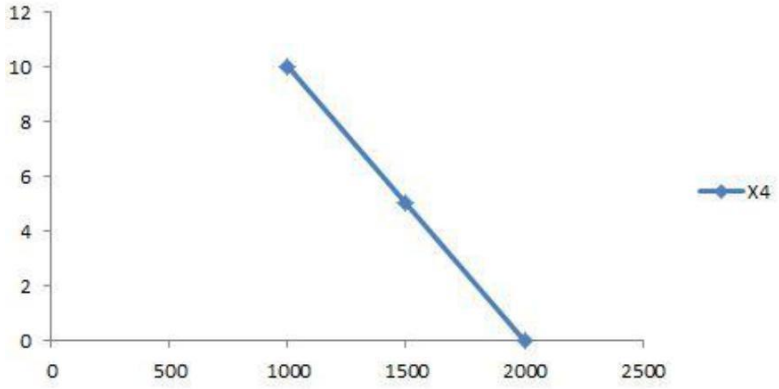

Рисунок 5.3 – X3, потенційний об‘єм програмного коду

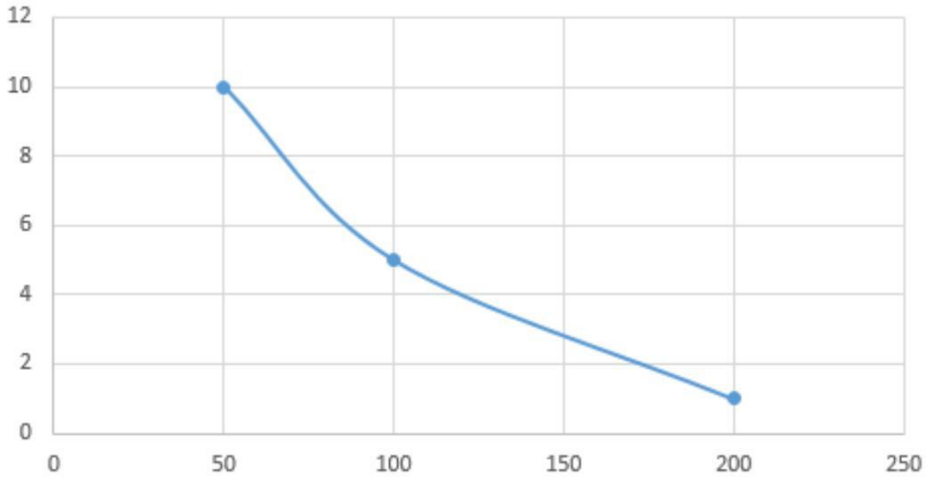

Рисунок 5.4 – X4,час обробки запитів користувача

5.4.3 Аналіз експертного оцінювання параметрів

Після детального обговорення й аналізу кожний експерт оцінює ступінь важливості кожного параметру для конкретно поставленої цілі – розробка програмного продукту, який має найбільш зручний інтерфейс та зрозумілу взаємодію з користувачем

Значимість кожного параметра визначається методом попарного порівняння. Оцінку проводить експертна комісія із 7 людей (табл. 5.3). Визначення коефіцієнтів значимості передбачає.

Таблиця 5.3 – Оцінки експертів

Позначення параметра	Назва параметра	Одиниці виміру	Ранг параметра за оцінкою експерта							Сума рангів R_i	Відхилення $_i$	^2_i
			1	2	3	4	5	6	7			
X1	Швидкодія мови програмування	Оп/мс	3	2	2	1	1	2	2	13	-4,5	20,25
X2	Об‘єм пам‘яті для збереження даних	Мб	1	1	1	2	2	1	1	9	-8,5	72,25
X3	Час обробки запитів користувача	Мс	2	3	3	3	3	3	4	21	3,5	12,25
X4	Потенційний об‘єм програмного коду	кількість строк коду	4	4	4	4	4	4	3	27	9,5	20,25
	Разом		10	10	10	10	10	10	10	70	0	195

Для перевірки степені достовірності експертних оцінок, визначимо наступні параметри:

а) сума рангів кожного з параметрів і загальна сума рангів:

$$R_i = \sum_{j=1}^{N} r_{ij} R_{ij} = \frac{Nn(n+1)}{2} = 70,$$

де N – число експертів, n – кількість параметрів;

б) середня сума рангів:

$$T = \frac{1}{n} R_{ij} = 17{,}5.$$

в) відхилення суми рангів кожного параметра від середньої суми рангів:

$$\Delta_i = R_i - T$$

Сума відхилень по всім параметрам повинна дорівнювати 0;

г)загальна сума квадратів відхилення:

$$S = \sum_{i=1}^{N} \Delta_i^2 = 195.$$

Порахуємо коефіцієнт узгодженості:

$$W = \frac{12S}{N^2(n^3 - n)} = \frac{12 \cdot 195}{7^2(4^3 - 4)} = 0{,}80 > W_k = 0{,}67$$

Ранжування можна вважати достовірним, тому що знайдений коефіцієнт узгодженості перевищує нормативний, котрий дорівнює 0,67.

Скориставшись результатами ранжирування, проведемо попарне порівняння всіх параметрів і результати занесемо у таблицю 5.4.

Таблиця 5.4 – Попарне порівняння параметрів

Параметри	Експерти							Кінцева оцінка	Числове значення
	1	2	3	4	5	6	7		
X1 і X2	>	<	<	<	>	<	<	<	0,5
X1 і X3	>	>	>	>	>	>	>	>	1,5
X1 і X4	>	>	>	>	>	>	>	>	1,5
X2 і X3	>	>	>	>	>	>	>	>	1,5
X2 і X4	>	>	>	>	>	>	>	>	1,5
X3 і X4	>	>	<	>	<	<	>	>	1,5

Числове значення, що визначає ступінь переваги *i*–го параметра над *j*–тим, a_{ij} визначається по формулі:

$$aij = \begin{cases} 1.5 \text{ при } Xi > Xj \\ 1.0 \text{ при Xi} = \text{Xj} \\ 0.5 \text{ при Xi} < Xj \end{cases}$$

З отриманих числових оцінок переваги складемо матрицю A=$\| a_{ij} \|$.

Для кожного параметра зробимо розрахунок вагомості $K_{вi}$ за наступними формулами:

$$K_{\text{вi}} = \frac{b_i}{\sum_{i=1}^{n} b_i}, \text{де } b_i = \sum_{i=1}^{N} a_{ij}.$$

Відносні оцінки розраховуються декілька разів доти, поки наступні значення не будуть незначно відрізнятися від попередніх (менше 2%).На

другому і наступних кроках відносні оцінки розраховуються за наступними формулами:

$$K_{\text{ві}} = \frac{b_i'}{\sum_{i=1}^{n} b_i'}, \text{де } b_i' = \sum_{i=1}^{N} a_{ij} b_j.$$

Як видно з таблиці 5.5, різниця значень коефіцієнтів вагомості не перевищує 2%, тому більшої кількості ітерацій не потрібно.

Таблиця 5.5 – Розрахунок вагомості параметрів

Параметриx_i	Параметриx_j				Перша ітер.		Друга ітер.		Третя ітер	
	X1	X2	X3	X4	b_i	$K_{\text{ві}}$	b_i^1	$K_{\text{ві}}^1$	b_i^2	$K_{\text{ві}}^2$
X1	1,0	0,5	1,5	1,5	4,5	0,281	16,25	0,275	59,125	0,274
X2	1,5	1,0	1,5	1,5	5,5	0,344	21,25	0,360	77,875	0,361
X3	0,5	0,5	1,0	1,5	3,5	0,219	12,25	0,208	44,875	0,207
X4	0,5	0,5	0,5	1,0	2,5	0,156	9,25	0,157	34,125	0,158
Всього:					16	1	59	1	216	1

5.5 Аналіз рівня якості варіантів реалізації функцій

Визначаємо рівень якості кожного варіанту виконання основних функцій окремо.

Абсолютні значення параметрів *X2*(об‘єм пам‘яті для збереження даних) та *X1* (швидкодія мови програмування) відповідають технічним вимогам умов функціонування даного ПП.

Абсолютне значення параметра *X3 (*потенційний об‘єм програмного коду) обрано не найгіршим (не максимальним), тобто це значення відповідає або варіанту а) 2000 або варіанту б) 1500

Абсолютне значення параметра *X4 (*час обробки запитів користувача) обрано не найкращим (не мінімальним), тобто це значення відповідає або варіанту а) 50 мс або варіанту б) 100 мс

Коефіцієнт технічного рівня для кожного варіанта реалізації ПП розраховується так (таблиця 4.6):

$$K_K(j) = \sum_{i=1}^{n} K_{вi,j} B_{i,j} ,$$

де n – кількість параметрів;$K_{вi}$– коефіцієнт вагомості i–го параметра; B_i – оцінка i–го параметра в балах.

За даними з таблиці 4.6 за формулою

$$K_K = K_{\text{ТУ}}[F_{1k}] + K_{\text{ТУ}}[F_{2k}] + \ldots + K_{\text{ТУ}}[F_{zk}],$$

визначаємо рівень якості кожного з варіантів:

K_{K1} = 1,37 + 2,42 + 1, 55 + 0,395 = 5,725

K_{K2} = 1,37 + 2,42 + 1,14 + 1,01 = 5,95

Таблиця 5.6 – Розрахунок показників рівня якості варіантів реалізації основних функцій ПП

Основні функції	Варіант реалізації функції	Параметри	Абсолютне значення параметра	Бальна оцінка параметра	Коефіцієнт вагомості параметра	Коефіцієнт рівня якості
F1	Б	X1	100	5	0,274	1,37
F2	А	X2	500	6,7	0,361	2.42
F3	А	X3	1500	2,5	0,158	0,395
	Б		800	6,4	0,158	1,01
	А	X4	50	7,5	0,207	1,55
	Б		100	5,5	0,207	1,14

Як видно з розрахунків, кращим є перший варіант, для якого коефіцієнт технічного рівня має найбільше значення.

5.5 Економічний аналіз варіантів розробки ПП

Програмний продукт, що буде встановлюватись на сервер кафедри вже розроблено, проте необхідно увесь рік тримати під контролем сервер, на якому буде встановлено програмне забезпечення. Отже

$T_1 = 90 \cdot 1.7 \cdot 0.8 = 122.4$ людино-днів.

$T_1 = 122.4 \cdot 8 = 979.2$ людино-годин;

В розробці беруть участь Unity3D програміст з окладом 14000 грн та фізик з окладом 11000 грн. Визначимо зарплату за годину за формулою:

$$С_ч = \frac{М}{T_m \cdot t} \text{ грн.},$$

де М – місячний оклад працівників; T_m – кількість робочих днів тиждень; t – кількість робочих годин в день.

$$С_ч = \frac{14000 + 11000}{2 \cdot 21 \cdot 8} = 74{,}4 \text{ грн.}$$

Тоді, розрахуємо заробітну плату за формулою

$$С_{ЗП} = С_ч \cdot Т_i \cdot К_Д,$$

де $С_ч$– величина погодинної оплати праці програміста;$Т_i$ – трудомісткість відповідного завдання; $К_Д$ – норматив, який враховує додаткову заробітну плату.

Зарплата розробників за варіантами становить:

$С_{ЗП} = 74{,}4 * 979.2 \cdot 1.2 = 72867.1$ грн на рік.

Відрахування на єдиний соціальний внесок незалежно від групи професійного ризику становить 22%:

$С_{ВІД} = С_{ЗП} \cdot 0.22 = 69900.6 * 0.22 = 16028.57$ грн.

Тепер визначимо витрати на оплату однієї машино-години. ($С_М$)

Працюватиме одна електронна обчислювальна машина цілодобово:

$С_Г = 12 \cdot М \cdot К_З = 12 \cdot 4000 \cdot 0{,}2 = 9600$ грн.

З урахуванням додаткової заробітної плати:

$$С_{ЗП} = С_Г \cdot (1 + К_З) = 9600 \cdot (1 + 0.2) = 11520 \text{ грн.}$$

Відрахування на єдиний соціальний внесок:

$$С_{ВІД} = С_{ЗП} \cdot 0.3666 = 11520 * 0{,}3666 = 4147 \text{ грн.}$$

Амортизаційні відрахування розраховуємо при амортизації 25% та вартості ЕОМ – 18000 грн.

$$С_А = К_{ТМ} \cdot К_А \cdot Ц_{ПР} = 1.15 \cdot 0.25 \cdot 18000 = 5175 \text{ грн.},$$

де $К_{ТМ}$– коефіцієнт, який враховує витрати на транспортування та монтаж приладу у користувача; $К_А$ – річна норма амортизації; $Ц_{ПР}$ – договірна ціна приладу.

Витрати на ремонт та профілактику розраховуємо як:

$$С_Р = К_{ТМ} \cdot Ц_{ПР} \cdot К_Р = 1.15 \cdot 18000 \cdot 0.05 = 1035 \text{ грн.},$$

де $К_Р$– відсоток витрат на поточні ремонти.

Ефективний годинний фонд часу ПК за рік розраховуємо за формулою:

$$Т_{ЕФ} = (Д_К – Д_В – Д_С – Д_Р) \cdot t_З \cdot К_В = (365 – 104 – 8 – 5) \cdot 8 \cdot 0.9 = 1785{,}6 \text{ годин},$$

де $Д_К$ – календарна кількість днів у році; $Д_В$, $Д_С$ – відповідно кількість вихідних та святкових днів; $Д_Р$ – кількість днів планових ремонтів устаткування; t –кількість робочих годин в день; $К_В$– коефіцієнт використання приладу у часі протягом зміни. Витрати на оплату електроенергії розраховуємо за формулою:

$$С_{ЕЛ} = Т_{ЕФ} \cdot N_С \cdot К_З = 1785{,}6 \cdot 0{,}6 \cdot 1.94 = 2078{,}43 \text{ грн.},$$

де N_C – середньо-споживча потужність приладу; $K_З$– коефіцієнтом зайнятості приладу; $Ц_{ЕН}$ – тариф за 1 КВт-годин електроенергії.

Накладні витрати розраховуємо за формулою:

$С_Н = Ц_{ПР} \cdot 0.67 = 18000 \cdot 0,67 = 12060$ грн.

Тоді, річні експлуатаційні витрати будуть:

$$С_{ЕКС} = С_{ЗП} + С_{ВІД} + С_А + С_Р + С_{ЕЛ} + С_Н$$

$С_{ЕКС} = 72867.1 + 4147 + 5175 + 1035 + 2078,43 + 12060 = 97362$ грн.

Собівартість однієї машино-години ЕОМ дорівнюватиме:

$С_{М\text{-}Г} = С_{ЕКС} / Т_{ЕФ} = 112638.12 / 1706,4 = 54,52$ грн/час.

Оскільки в даному випадку всі роботи, які пов‘язані з розробкою програмного продукту ведуться на ЕОМ, витрати на оплату машинного часу, в залежності від обраного варіанта реалізації, складає:

$$С_М = С_{М\text{-}Г} \cdot Т$$

$С_М = 66 * 979.2 = 53392$

Накладні витрати складають 67% від заробітної плати:

$$С_Н = С_{ЗП} \cdot 0,67$$

$С_Н = 69900.6 * 0.67 = 48244.1$ грн.;

Отже, вартість розробки ПП за варіантами становить:

$$С_{ПП} = С_{ЗП} + С_{ВІД} + С_М + С_Н$$

$С_{ПП} = 72867 + 4147 + 53392 + 48244.1 = 178650,1$ грн.;

5.6 Вибір кращого варіанта ПП техніко-економічного рівня

Розрахуємо коефіцієнт техніко-економічного рівня за формулою:

$$К_{ТЕРj} = К_{Кj} / С_{Фj},$$

$$К_{ТЕР1} = 5,325 / 178650,1 = 0,56 * 10^{-4};$$

$$К_{ТЕР2} = 6,35 / 178650,1 = 0,35 * 10^{-4};$$

Як бачимо, найбільш ефективним є перший варіант реалізації програми з коефіцієнтом техніко-економічного рівня $К_{ТЕР1} = 5.6 \cdot 10^{-5}$.

5.7 Висновки

В даному розділі проведено повний функціонально-вартісний аналіз ПП, який було розроблено в рамках дипломного проекту. Процес аналізу можна умовно розділити на дві частини.

В першій з них проведено дослідження ПП з технічної точки зору: було визначено основні функції ПП та сформовано множину варіантів їх реалізації; на основі обчислених значень параметрів, а також експертних оцінок їх важливості було обчислено коефіцієнт технічного рівня, який і дав змогу визначити оптимальну з технічної точки зору альтернативу реалізації функцій ПП.

Другу частину ФВА присвячено вибору із альтернативних варіантів реалізації найбільш економічно обґрунтованого. Порівняння запропонованих варіантів реалізації в рамках даної частини виконувалось за коефіцієнтом ефективності, для обчислення якого були обчислені такі допоміжні параметри, як трудомісткість, витрати на заробітну плату, накладні витрати.

Після виконання функціонально-вартісного аналізу програмного комплексу що розроблюється, можна зробити висновок, що з альтернатив, що

залишились після першого відбору двох варіантів виконання програмного комплексу оптимальним є перший варіант реалізації програмного продукту. У нього виявився найкращий показник техніко-економічного рівня якості

КТЕР = 0,56

Цей варіант реалізації програмного продукту має такі параметри:
Мова програмування C#;
Unity3D як ігровий рушій;
Тематика ВЛ - Фізична

Отже обраний варіант створення демонстраційного прототипу найкраще підійде для ілюстрації можливостей обраних технологій.

ВИСНОВКИ

В ході даної дипломної роботи, в першому розділі було досліджено основні види та характеристики моделей. Були освітлені основні принципи та процеси моделювання. Також була сформулювана актуальність роботи та визначенні задачі.

В другому розділі було розглянуто найбільш популярні сучасні засоби для створення віртуальних лабораторій. Проаналізовано їх базові заявлені функціональні можливості.

Серед ігрових рушіїв у колі незалежних розробників, останнім часом, найбільшою популярністю користуется Unity. Він повністью задовольняє нас за своїми базовими характеристиками до тогож має потужну підтримку товариства, що значно знижує поріг входження.

Unity – ігровий рушій, зібравший в собі фізичний, графічний рушій, великий набір інструментів та широкий функціонал. Гарний баланс міл кількістю написаного коду та використання вбудованого інструментарію для налаштування елементів гри. Зручна візуалізація продукту та чудові можливості відладки як коду так і поведінки гри в цілому. Великі можливості для повторного використання коду та вбудована оптимізація графіки, фізики. Unity виявився найпотужнішим засобом для розробки ВЛ серед розглянутих, функціонал двигуна ідеально задовольняє основні вимоги до завдання створення віртуальної лабораторії. Найкраще Unity покаже себе в розробці великих та середніх проектів командою.

ПЕРЕЛІК ПОСИЛАНЬ

1. Дубовой В. М. Ідентифікація та моделювання технологічних об’єктів і систем керування : навчальний посібник / В. М. Дубовой. – Вінниця : ВНТУ, 2012. – 308 с.
2. Маликов В. Т. Анализ измерительных информационных систем / Маликов В. Т., Дубовой В. М., Кветный Р. Н., Исматуллаев П. Р.Ташкент: ФАН, 1984. – 176 с.
3. Chris Richardson. From Design to Deployment / Chris Richardson, Floyd Smith, 2016. – 74 p.
4. Офіційний сайт Unity. – Режим доступу: https://unity3d.com/ – Дата доступу: 27.05.2017
5. MIT iCampus iLabs. – Режим доступу: http://icampus.mit.edu/projects/ilabs/ – Дата доступу: 27.05.2017
6. William Sherif Learning C++ by Creating Games with UE4, 2016. – 74 p.
7. Офіційний сайт Turbulenz. – Режим доступу: http://biz.turbulenz.com/ – дата доступу: 11.03.2017
8. Офіційний сайт Google Cardboard. – Режим доступу: https://vr.google.com/cardboard/ – дата доступу: 16.05.2017
9. Матеріали сайту Інфотехно – Сравнительная характеристика систем дистанционного обучения [Електронний ресурс] – Режим доступу: http://www.infotechno.ru/analizSDO.htm – дата доступу: 17.05.2017