Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина)
(СПбГЭТУ «ЛЭТИ»)

Направление	09.03.04 Программная инженерия
Профиль	Разработка программно-информационных систем
Факультет	КТИ
Кафедра	МО ЭВМ

К защите допустить

Зав. кафедрой Кринкин К.В.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА БАКАЛАВРА

ТЕМА: РАЗРАБОТКА 3D VR-ИГРЫ «САПЁР»

Студентка _____ Горбунова А.
подпись

Руководитель к.т.н., доцент _____ Романцев В.В.
подпись

Консультанты к.э.н., доцент _____ Антонова И.М.
подпись

к.т.н. _____ Заславский М.М.
подпись

Санкт-Петербург
2020

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ

Утверждаю
Зав. кафедрой МО ЭВМ
_____ Кринкин К.В.
«___»_____2020 г.

Студентка Горбунова А. Группа 6303
Тема работы: Разработка 3D VR-игры «Сапёр»

Место выполнения ВКР: Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина)

Исходные данные (технические требования): Разработать приложение, реализующее игровой процесс «Сапёра» в трехмерном пространстве виртуальной реальности на базе одной из сред разработки компьютерных игр.

Содержание ВКР:
Аналитический обзор современного состояния вопроса, Проектирование ПО, Реализация ПО, Анализ результатов, Экономическое обоснование ВКР.

Перечень отчетных материалов: пояснительная записка, иллюстративный материал.

Дополнительные разделы: Экономическое обоснование ВКР.

Дата выдачи задания «27» апреля 2020 г.

Дата представления ВКР к защите «24» июня 2020 г.

Студентка _____ Горбунова А.

Руководитель к.т.н., доцент _____ Романцев В.В.

КАЛЕНДАРНЫЙ ПЛАН ВЫПОЛНЕНИЯ
ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ

Утверждаю
Зав. кафедрой МО ЭВМ
_____ Кринкин К.В.
«___»_____2020 г.

Студентка Горбунова А. Группа 6303
Тема работы: Разработка 3D VR-игры «Сапёр»

№ п/п	Наименование работ	Срок выполнения
1	Обзор литературы по теме работы	27.04 – 29.04
2	Обзор аналогов и выбор средств разработки	30.04 – 02.05
3	Проектирование ПО	03.05 – 07.05
4	Реализация ПО	08.05 – 15.05
5	Тестирование и отладка	16.05 – 18.05
6	Анализ полученных результатов	19.05 – 21.05
7	Подготовка экономико-технического обоснования	22.05 – 23.05
8	Оформление пояснительной записки	23.05 – 31.05
9	Оформление иллюстративного материала	01.06 – 05.06
10	Предзащита	03.06 – 18.06
11	Получение допуска к защите	19.06 – 21.06
12	Защита	24.06

Студентка _____ Горбунова А.

Руководитель к.т.н., доцент _____ Романцев В.В.

РЕФЕРАТ

Пояснительная записка 77 стр., 32 рис., 13 табл., 16 ист.

ВИРТУАЛЬНАЯ РЕАЛЬНОСТЬ, РАЗРАБОТКА ИГР, САПЁР, VR-ИГРЫ, UNREAL ENGINE 4.

Объектом исследования данной работы являются VR-игры.

Предметом исследования работы являются технологии организации пользовательского взаимодействия и перемещения пользователя в пространстве виртуальной реальности.

Цель работы - разработка трехмерной VR-версии игры «Сапёр» на базе одной из сред разработки компьютерных игр.

В рамках настоящей работы разработаны решения по переносу двухмерного игрового мира в пространство виртуальной реальности, а также по организации пользовательского интерфейса и перемещения пользователя в этом пространстве. В процессе выполнения работы были также разработаны алгоритмы реализации игровой логики «Сапера» в трехмерном представлении. В результате выполнения данного проекта было получено полноценное игровое приложение, отличающееся удобством и простотой взаимодействия пользователя с виртуальной средой.

ABSTRACT

The aim of the work is to develop a three-dimensional VR-version of the game "Minesweeper" based on one of the development environments for computer games.

In this work, solutions were developed on transferring a two-dimensional game world into a virtual reality space, as well as on organizing a user interface and moving a user in this space. In the process of doing the work, algorithms for the implementation of the Sapper game logic in a three-dimensional representation were also developed.

As a result of the implementation of this project, a full-fledged game application was obtained, characterized by the convenience and simplicity of user interaction with the virtual environment.

СОДЕРЖАНИЕ

ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

В настоящей пояснительной записке применяются следующие сокращения и термины с соответствующими определениями:

HMD – Head Mounted Display – устройство реализации виртуальной реальности, крепящееся на голову, часто представлено в виде очков или шлема.

HUD – Heads-Up Display – часть пользовательского интерфейса в игре, располагающаяся на переднем плане игрового пространства.

UI – User Interface (Пользовательский интерфейс)

VR (ВР) – Virtual Reality (Виртуальная реальность)

Аватар – графическое изображение главного игрового персонажа.

Ассет – игровой объект в Unreal Engine 4

Актер – ассет, помещенный на уровень

Виджет – визуальное представление элементов UI.

ВКР – выпускная квалификационная работа.

1. ВВЕДЕНИЕ

В настоящее время индустрия компьютерных игр огромна по своим масштабам и все еще продолжает расти. На рынке игр уже сейчас можно подобрать продукт практически под любой запрос. Однако игровая аудитория все еще расширяется, появляются как новые игры уже существующих жанров, так и новые игровые жанры, улучшается графика, производительность, создаются новые платформы для запуска игр, привлекаются новые технологии, расширяется возрастной диапазон игроков.

Одним из жанров, охватывающих практически всю игровую аудиторию, является казуальный жанр. Особенностью казуальных игр является отсутствие в них сложных правил и сюжета, они просты в управлении, не требуют усидчивости и долгого обучения, а их разработка, как правило, не занимает много времени и ресурсов. Каждый, кто знаком с компьютерными играми, хотя бы раз встречался с казуальным жанром. Распространенным примером казуального жанра является игра «Сапёр». Изначально «Сапёр» появился в формате настольной игры, но свою широкую популярность приобрел с момента появления персональных компьютеров. Существует много версий этой игры-головоломки, отличающихся разным оформлением, различными модификациями правил и представлением поля игры.

Развитие игровой индустрии и привлечение новой аудитории происходит не только за счет увеличения количества приложений для существующих платформ, но и за счет использования новых технологий в разработке и создания новых платформ для запуска игр. Сегодня компьютерная техника достигла настолько высокого уровня развития, что позволяет разрабатывать такие реалистичные игры, что пользователи ощущают себя частью игрового мира. Это достигается за счет воздействия техническими средствами на органы чувств человека: зрение, слух, осязание и другие, и называется виртуальной реальностью. Технические средства

имитируют воздействия на человека во время игрового процесса и его реакции на это воздействие.

Несмотря на то, что это направление молодое для игрового мира, оно стремительно набирает популярность и имеет большие перспективы. Для VR-платформ создано огромное количество новых продуктов, которыми игроки уже активно пользуются. Однако игры для персонального компьютера не утратили своей популярности. Данная работа посвящена решению задачи переноса двумерной компьютерной игры в трехмерное игровое пространство с использованием технологий виртуальной реальности.

1.1. Постановка задачи

В ходе выполнения данной работы необходимо спроектировать и разработать приложение, обеспечивающее поддержку следующих функциональных возможностей:

1. Организация игрового процесса, соответствующего игре «Сапёр»
2. Представление игрового мира в пространстве ВР
3. Восприятие происходящего пользователем глазами игрового персонажа
4. Реализация перемещения в пространстве ВР
5. Реализация пользовательского взаимодействия посредством ВР-устройств ввода

1.2. Цель, задачи, объект и предмет исследования

Цель работы: разработка трехмерной VR-версии игры «Сапёр» на базе одной из сред разработки компьютерных игр.

Для достижения поставленной цели должны быть решены следующие задачи:

1. Провести обзор, сравнительный анализ и определить программные средства разработки
2. Спроектировать архитектуру приложения

3. Разработать алгоритм реализации игрового процесса
4. Описать подходы к организации перемещения пользователя в пространстве ВР
5. Спроектировать и реализовать разрабатываемый программный модуль на базе выбранных средств разработки
6. Провести отладку и тестирование разработанного приложения.

Объектом исследования данной работы являются VR-игры.

Предметом исследования работы являются технологии организации пользовательского взаимодействия и перемещения пользователя в пространстве виртуальной реальности.

1.3. Новизна и практическая значимость работы

Научная новизна работы заключается в том, что в работе описаны методы переноса игрового процесса и игрового мира из двумерного пространства в пространство виртуальной реальности.

Практическая значимость состоит в том, что описанные методы позволят применять их к другим двумерным играм и приложениям, для которых будет стоять задача переноса в ВР-пространство. Это позволит внести большой вклад в индустрию компьютерных игр, использующих технологии ВР и вернуть актуальность играм, которые ее стремительно теряют с приходом новых XR-технологий.

Практическая новизна работы заключается в создании уникального продукта, позволяющего игровой аудитории, предпочитающей известную игру «Сапёр», сыграть в нее в новом формате ВР.

Методы разработки, описанные в данной работе могут быть использованы в других проектах, что может привести к сокращению сроков разработки новых ВР-продуктов, имеющих двумерные версии.

2. АНАЛИТИЧЕСКИЙ ОБЗОР СОВРЕМЕННОГО СОСТОЯНИЯ ВОПРОСА

2.1. Анализ и общая характеристика предметной области

До начала научно-технической революции представление виртуальной реальности существовало лишь в качестве научно-фантастической идеи. Но под виртуальностью уже тогда понимали объект или состояние, которые реально не существуют, но могут возникнуть при определенных условиях [1].

В качестве посредника между пользователем и виртуальным миром выступают VR-устройства ввода-вывода, обеспечивающие взаимодействие пользователя с виртуальной средой (рисунок 2.1). Устройства вывода воздействуют на органы чувств пользователя, имитируя его присутствие в виртуальном мире, устройства ввода улавливают реакцию на это воздействие и передают ее обратно в ВР.

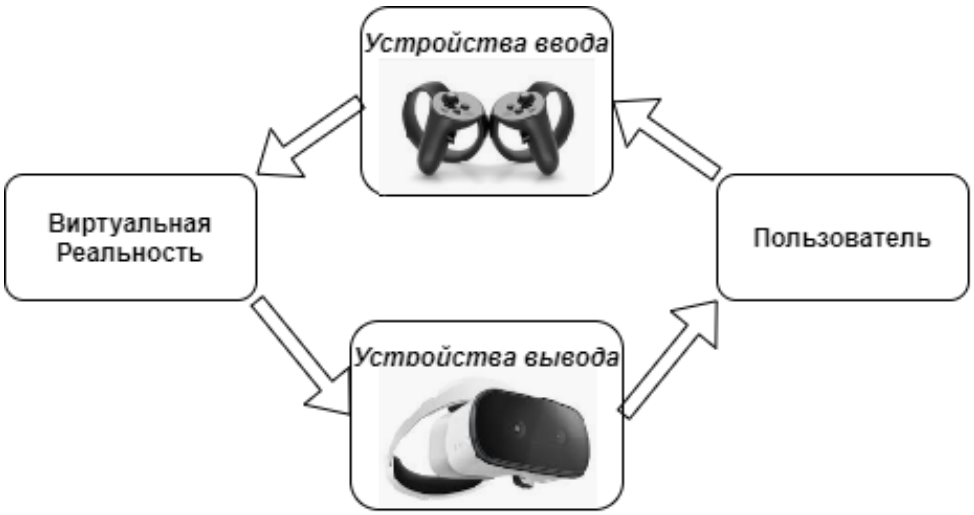

Рисунок 2.1 – Взаимодействие пользователя с ВР

Самыми первыми предпосылками к созданию таких устройств можно выделить стереоскопы, появившиеся еще в начале 19 века. С помощью этих, в то время примитивных, устройств, работающих по принципу стереоскопического зрения, пользователи смогли «погружаться» в изображение.

В 1957 году (устройство было запатентовано в 1962 году [2]) американский инженер и кинорежиссер Мортон Хайлиг создает первый в мире виртуальный симулятор – сенсораму. Она представляет собой одноместную кабину, в которой транслировался короткий фильм, снятый от первого лица. При этом кабина симулировала воздействие и на другие органы чувств – устройство было оснащено вентиляторами, генераторами запахов, стереодинамиками и вибрационным сиденьем.

Спустя всего 6 лет в 1968 году американский ученый Айвен Свазерленд создал устройство, которое напоминало современный VR-шлем. Оно было довольно массивным, так что приходилось крепить его к потолку, но несмотря на это изобретение было довольно прогрессивным для того времени. Оно могло отображать простейшие геометрические формы, а также отслеживать положение пользователя в пространстве благодаря магнитным датчикам, расположенным на потолке.

Термин «виртуальная реальность» был введен в 1987 году основателем лаборатории виртуального программирования Джоран Ланье. Его лаборатория выпустила костюм для виртуальной реальности, который состоял из самого костюма с датчиками, которые могли отслеживать положение рук и ног, первой в своем роде перчатки-контроллера [3] и шлема. Именно это изобретение привело к рассвету ВР в середине 90-х годов. Тогда виртуальная реальность показалась привлекательной сразу двум гигантам игровой индустрии – SEGA и Nintendo. Но высокая цена, маленький список доступных игр и неудобство использования представленных этими компаниями устройств привели к тому, что широкого распространения они так и не получили. Эти неудачи повлияли на рынок виртуальной реальности. Устройства продолжали выпускаться, но они были интересны скорее отдельным любителям, но не массовому пользователю.

В 2011 году большой любитель виртуальной реальности - 18-летний Палмер Лаки собрал первый прототип виртуального шлема, который в дальнейшем получит название Oculus Rift. В 2012 году была основана

компания Oculus VR, а в 2013 году вышла первая версия Oculus Rift [4] под названием DK1, которая вернула интерес аудитории к виртуальной реальности. Успех компании Oculus VR привлек внимание компаний-гигантов игровой индустрии. Вскоре вышли на рынок такие устройства как HTC Vive, PlayStation VR, Google Cardboard, Gear VR и другие.

Кроме того, в настоящее время VR-шлемы являются только одним из вариантов аппаратных средств реализации виртуальной реальности. В качестве еще одного варианта можно назвать MotionParallax3D-дисплеи. Это тип VR-устройств, которые создают имитацию объемного объекта за счет отображения на одном или нескольких дисплеях проекцию этого виртуального объекта в зависимости от положения глаз пользователя. Существует много вариантов реализации такой системы: от смартфонов до виртуальных комнат. Но VR-шлемы отличаются тем, что полностью изолируют пользователя от реального мира, а MotionParallax3D-дисплеи в той или иной степени имеют ограниченную среду воздействия, так как пользователь видит как виртуальные объекты, так и реальные. Это снижает уровень погружения пользователя в виртуальную среду.

Еще один вариант реализации VR-системы – виртуальный ретинальный монитор. Его идея состоит в том, что изображение проектируется прямо на сетчатку глаза пользователя. Данный тип устройств ближе к системам дополненной реальности, так как пользователь видит виртуальные объекты на фоне реальных, а полное погружение достигается лишь при определенных условиях (достаточно большое и графически-плотное проектируемое изображение, использование устройства в темном помещении и т.д.). Существуют и другие VR-устройства, например, имитирующие тактильные ощущения и даже обеспечивающие прямое воздействие на нервные окончания пользователя. Но к настоящему времени они не получили широкого распространения у массового пользователя, а многие из них и вовсе существуют в виде прототипов.

Самыми распространенными VR-устройствами у массового использования являются шлемы, которые также получили название HMD или очки виртуальной реальности. Это конструкция, оснащенная дисплеем и акустической системой, которая крепится на голову пользователя. Изображение на дисплее транслируется для каждого глаза отдельно, по принципу стереоскопа, создавая объемное изображение. Датчики на таком устройстве отслеживают положение головы и транслируют соответствующее ему изображение на дисплей. В качестве дисплея также может быть использован дисплей смартфона (Google Cardboard, Samsung Gear VR).

Шлемы виртуальной реальности находят широкое применение в самых различных сферах деятельности человека. Они используются для отображения пространства в условиях плохой видимости и низкой освещенности, для объемного отображения при проектировании и изучении сложных систем, для обучения в условиях отсутствия для этого реальных условий.

К настоящему времени на рынке появилось огромное количество HMD, имеющих различный функционал и, соответствующую ему цену. Более дорогие версии включают в себя более качественные дисплеи и различные виды датчиков, отслеживающих положение пользователя, бюджетные версии предоставляют ограниченный функционал (например, VR-очки для смартфонов). Развитие этой индустрии привело к тому, что устройства виртуальной реальности стали доступны для массовой аудитории. Одновременно с этим появился спрос на контент. Появилось ПО, поддерживающее отображение информации в формате ВР. Наиболее востребованным контентом у массового пользователя в этой сфере являются компьютерные игры. Это связано с тем, что игровой процесс основан на взаимодействии пользователя с игровым миром. Чем качественнее и реалистичнее представлен этот мир, тем больше уровень погружения в него пользователя за счет приближения игрового мира к реальному. Также многие игры представлены в формате от первого лица, где игрок отождествляется с

главным персонажем, непосредственно участвующим в игровом процессе. HMD включают все эти функциональные возможности. Пользователь испытывает полное погружение: теперь он видит объемную трехмерную картинку вместо плоского дисплея монитора, может совершать движения и повороты головы, как и его игровой персонаж, вместо нажатия клавиш мыши или клавиатуры.

На новой волне интереса к ВР появилась проблема систем ввода. Погружению в виртуальный игровой мир может мешать несоответствие команд пользовательского интерфейса осуществляемым в игре действиям. Также, привычное управление мышью и клавиатурой может оказаться бесполезным в некоторых вариантах VR-игр, а управления наведением взгляда может быть недостаточно. Так как при использовании HMD пропадает зрительная связь с реальным миром, то использование сложных комбинаций на клавиатуре будет мешать игровому процессу. А если для управления требуется поворачиваться или передвигаться в реальности, то такое управление и вовсе станет невозможным. Чтобы избавиться от этой проблемы, были созданы различные игровые манипуляторы. Например, игровой компьютерный руль с педалями, целеуказатель оружия, джойстик или контроллеры. Последние могут предложить самые широкие возможности для систем ввода. Контроллеры чувствительны к движению и оснащены датчиками, отслеживающими их положение в пространстве, являются практически универсальным устройством управления для любой тематики и жанра VR-игры. Управление с помощью них позволяет обеспечить естественное и интуитивно понятное взаимодействие с виртуальной средой.

С ростом популярности VR-устройств на массовом рынке, крупные компании-разработчики обратили свое внимание на создание для них контента. Аналитики из исследовательского центра SuperData прогнозируют, что мировой доход рынка программного обеспечения для ВР-сегмента в 2020 году составит $2,6 млрд, а уже к 2023 году этот показатель вырастет до $5,1 млрд [5].

Большинство компьютерных игр, поддерживающих ВР, создается целенаправленно для использования с VR-устройствами. В этом случае разработчики при проектировании процесса и структуры игры стараются максимально использовать все возможности этих устройств. И такой формат игр стремительно набирает популярность у пользователей, однако игры, представленные для ПК, мобильных и консольных платформ не сдают позиции. Эта сфера также продолжает развиваться и расширяться, компании-разработчики ежедневно поставляют на игровой рынок огромное количество новых качественных продуктов. У многих существующих ПК-игр есть великое множество поклонников среди пользователей. Перевыпуск таких игр для VR-платформ позволит повысить интерес к продукту (или вернуть его, если интерес был снижен), расширить пользовательскую аудиторию, предоставив ей выбор удобного формата для каждого, увеличить доходы от продукта. В рамках данной работы представлена реализация версии двухмерной компьютерной игры «Сапёр» для ВР-платформы.

2.2. Обзор существующих решений

«Сапёр» - это компьютерная игра, относящаяся к жанру казуальные игры и жанру головоломки. Эта игра имеет огромное количество вариантов реализации, но всех их объединяет идея и основные принципы игры. Игровое поле разделено на смежные ячейки. Некоторое количество ячеек содержит мины, их количество известно пользователю. В начале игры все ячейки закрыты. Игрок по одной открывает ячейки, стараясь не попасть на ячейку с миной. Если на открытой ячейке мины не было, на ней появляется цифра, отражающая, сколько мин находится по соседству с открытой ячейкой. Используя эти числа, игрок старается рассчитать расположение всех мин на игровом поле. Иногда, ячейки приходится открывать наугад. Если соседние ячейки при открытии тоже не содержат мин, то открывается вся часть поля до ячеек, содержащих мины. Игрок может пометить ячейки, в которых, как он думает, находятся мины. Цель игры – открыть все ячейки на игровом

поле, не содержащие мин. Как правило, в игре представлено несколько уровней сложности.

В большинстве вариантов реализации есть подсчет времени прохождения игры. «Сапёр» получил такое распространение, что сейчас проводятся целые турниры по стандартным уровням сложности, а также регистрируются мировые рекорды времени их прохождения.

Каноническим вариантом реализации и самой распространенной версией игры «Сапёр» является стандартная игра ОС Windows, которая вышла с версией Windows 3.1 в 1992 году. При упоминании «Сапёра» любому, кто когда-либо являлся пользователем Windows, скорее всего придет на ум именно эта версия игры. Поле в ней представлено в виде прямоугольника, состоящего из ячеек-квадратов серого цвета (рисунок 2.2). Игроку предлагается 3 уровня сложности: новичок (9x9 ячеек, 10 мин), любитель (16x16 ячеек 40 мин), профессионал (30x16 ячеек, 99 мин) и пользовательский уровень (с возможностью самостоятельно выбрать размер поля и количество мин). «Сапёр» от Windows отличается от других версий тем, что имеет интерактивное игровое поле. Если все возможные комбинации, которые могут быть рассчитаны игроком, открыты, то при нажатии наугад в любом участке поля, мины на нем не окажется. Особенностью этой версии является минималистичный дизайн, примитивная графика, простота управления, встроенный в окно приложения пользовательский интерфейс. Компания Microsoft впоследствии сообщила, что это приложение было создано для более быстрого освоения пользователями работы с кнопками компьютерной мыши. Поэтому усложнения правил и красочного интерфейса оно не требовало, игра и без этого выполняла свою функцию. С выходом новых версий ОС Windows игра «Сапёр» также потерпела изменения и сейчас вовсе не похожа на свою первоначальную версию, однако в рамках настоящей работы рассматривается ее первая версия.

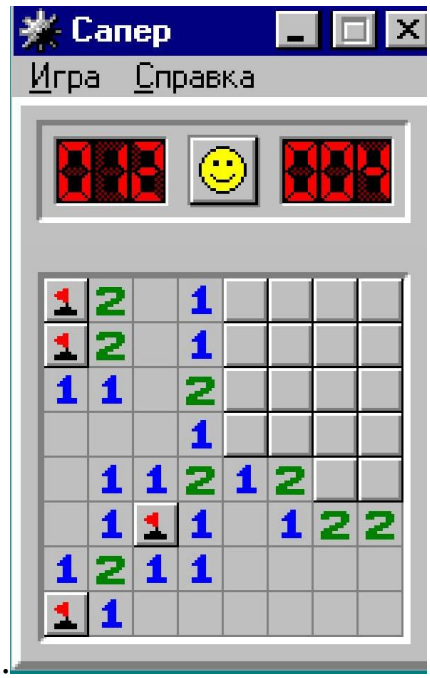

Рисунок 2.2 - Скриншот стандартной игры «Сапёр» от Windows

Существуют различные варианты игры «Сапёр» с разными геометрическими формами поля и ячеек в нем, варианты с разным определением «соседства» этих ячеек, многопользовательские варианты. В большинстве этих вариантов игровое поле представлено в двумерном пространстве. Одной из задач данной работы является «перенос» двумерного «Сапёра» в трехмерный игровой мир, поэтому был проведен поиск подобных реализаций этой игры, результат, которого представлен далее.

Tilesweeper - игра для ПК, разработанная IEVO в 2018 году, один из вариантов реализации «Сапёра», но с 3D-графикой (рисунок 2.3). Ячейки представлены в виде плитки, составляющих пол в трехмерном интерьере. При открытии притки, она трескается и проваливается вниз. Как такового аватара в игре нет, пользователь смотрит на игровой процесс сверху. Для каждого уровня представлены разные интерьеры и два типа поля – классический и аркадный. Классический тип поля – это поля стандартных размеров: 9x9, 16x16, 16x30. Аркадный тип – поля нестандартных форм и размеров. По правилам и принципам игры эта версия от версии Windows практически не отличается. Главным ее отличием является внешний вид:

трехмерное представление поля, живая графика, визуальные и аудио-эффекты.

Рисунок 2.3 - Скриншот игры Tilesweeper

Crazy Sapper 3D - еще одна версия «Сапёра» от разработчика Aratog LLC, вышедшая в 2016 году. Игра представлена для ПК и для мобильных устройств. В отличии от рассмотренных ранее версий, в этой присутствует продуманный сюжет, о котором рассказывается в начале игры и представлено на официальном сайте в качестве описания игры. В нем говорится о том, что генерал в отставке по имени Борис выкрал разрушительную вакцину, с помощью которой он планирует изготовить смертельную для всего мира бомбу, и спрятал ее на секретной базе, окруженной минными полями. Игроку предлагается почувствовать себя в роли сапёра по имени Макс, который должен найти вакцину и спасти мир. Поле игры представлено в виде «полосы препятствий» - прямоугольника, состоящего из ячеек (рисунок 2.4). В этой версии игры появляется аватар, внешний вид которого можно изменять. В начале игры аватар появляется в начале полосы, пользователь наблюдает за игровым процессом сверху от третьего лица. Цель игры – дойти до финиша, располагающегося в конце поля, то есть открывать все ячейки не обязательно, что противоречит классическому варианту. Но принцип открытия ячеек остался прежним, как и

возможность помечать ячейки флажком, за исключением того, что теперь можно открывать только смежные с уже открытыми ячейки. В этой версии игроку предлагается большое количество дополнительных способностей: у аватара имеется 3 жизни, появилась возможность срезать деревья, встречающиеся ему на пути, использовать миноискатель, который открывает несколько ячеек сразу, использовать динамит, проверяющий наличие мины на ячейке без потери жизни аватара и другие. Эти способности можно приобретать за игровые деньги, которые зарабатываются путем выполнения заданий. Crazy sapper 3D сильно отличается от рассмотренных ранее версий игры не только наличием дополнительных способностей и сюжета, но и наличием 30 различных уровней с нарастающей сложностью и внутриигрового магазина, в котором можно приобретать способности, аватаров и так далее. Но главным отличием является наличие аватара, которого не было в ранее рассмотренных версиях. Игровой мир представлен в трехмерном пространстве, как и в случае с Tilesweeper, но за счет появления персонажа на игровом поле с возможностью управлять им, усиливается ощущение присутствия и погружения в игру. Но игровой процесс ведется от третьего лица, что уменьшает уровень погружения.

Рисунок 2.4 - Скриншот игры Crazy Sapper 3D

Результаты анализа и сравнения найденных аналогов были сведены в таблицу 2.1.

Таблица 2.1 – Сравнение аналогов приложения

Критерий сравнения	Tilesweeper	Crazy Sapper 3D
Представление игрового мира	3D	3D
Наличие аватара	-	+
Вид	От третьего лица	От третьего лица
Изменения в правилах игры	-	+
Наличие стандартных уровней	+	-

Найденные версии игры «Сапёр» реализованы в трехмерном игровом мире как с аватаром с видом от третьего лица, так и без аватара вовсе. Его наличие усиливает ощущение присутствия и погруженности в игровой мир, но не создает полного погружения, так как пользователь все еще смотрит в плоский дисплей и воспринимает происходящее только с того ракурса, с которого установил разработчик игры. Таким образом, по итогам обзора не найдено ни одной версии, реализованной в пространстве ВР, а также не найдены версии, реализовывающие вид от первого лица. Следовательно, разрабатываемое в данной работе приложение является уникальным.

Выводы

Таким образом, в данном разделе ВКР была изучена предметная область: проанализировано современное состояние игровой индустрии, а именно ее VR-сегмента. Были исследованы аппаратные средства реализации виртуальной реальности. А также был выполнен поиск аналогов разрабатываемого приложения. По результатам поиска, были найдены только игровые приложения, реализующие игровую логику «Сапёра» в трехмерном пространстве от третьего лица.

Благодаря разрабатываемому приложению поклонники известной игры смогут сыграть в нее в новом формате. Кроме того, что виртуальная реальность обеспечивает уровень погружения, которого невозможно достичь с помощью других устройств, она способствует развитию у человека таких когнитивных навыков как: пространственное воображение, кратковременная память и внимание [6]. Игровой процесс, основанный на принципах «Сапёра», способствует развитию логического мышления и быстроты реакции.

3. ПРОЕКТИРОВАНИЕ ПО

3.1. Сравнительный анализ и обоснование выбора средств разработки

С ростом популярности VR-технологии растет количество специалистов в этой сфере и разнообразие средств создания подобных игр. Под средствами создания понимаются среды разработки компьютерных игр (игровые движки), которые представляют собой набор визуальных инструментов для разработки. Игровой движок и устройства ввода-вывода выступают в роли аппаратного обеспечения для моделирования ВР (рисунок 3.1).

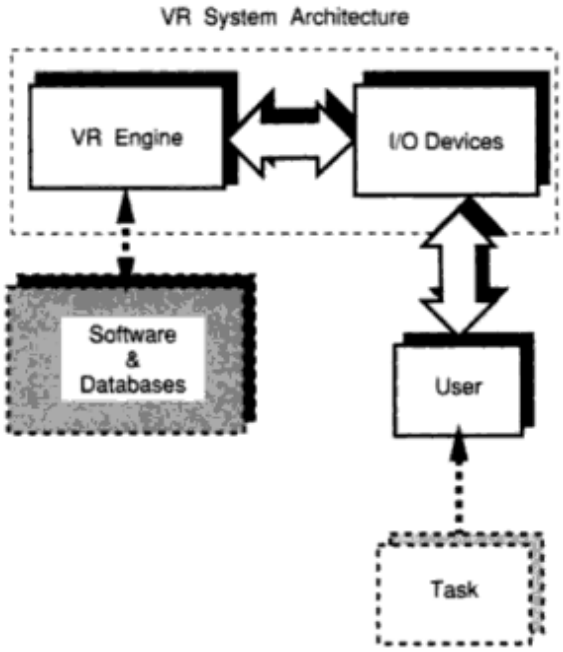

Рисунок 3.1 – Архитектура VR-системы

Источник: [7]

Правильно подобранный игровой движок упрощает процесс разработки игр и сокращает время разработки. Учитывая разнообразие существующих вариантов, возникает проблема выбора наиболее подходящей для конкретного проекта и условий разработки.

В настоящее время рынок игровых движков настолько насыщен, что может предложить решение для каждого разработчика, у которого может

быть различный уровень владения языками программирования, размер проектного бюджета, цели проекта и т.д. Для того чтобы провести сравнительный анализ между средами разработки, необходимо выявить факторы, влияющие на выбор разработчиком той или иной среды, они и будут впоследствии являться критериями сравнения.

Одним из ключевых факторов при выборе среды разработки является цена. Она зависит от бюджета и целей проекта. Как правило, начинающие разработчики довольствуются бесплатными или недорогими персональными версиями, а коммерческие организации чаще могут себе позволить приобрести платные, более дорогие версии с расширенным функционалом. На цену при выборе движка внимание обращается практически во всех случаях. При выборе по этому критерию предпочтение отдается более дешевым продуктам.

Еще одним фактором, влияющим на выбор движка, является поддерживаемая операционная система. Среда разработки должна обязательно поддерживаться операционной системой разработчика, если среда не поддерживается, то она для него недоступна.

Следующим фактором является целевая платформа. Под целевой платформой подразумевается сочетание оборудования и программного обеспечения, которое определяет конкретную среду выполнения разработанной игры. Целевая платформа проекта определяется его требованиями.

Операционная система и целевая платформа одинаково важны для каждого проекта и каждого разработчика. Чем больше операционных систем и целевых платформ поддерживает среда разработки, тем больше проектов может быть в ней выполнено.

Не менее важны при выборе языки программирования, на которых предполагается разработка в среде. Они являются основными инструментами в создании игр и приложений. Процесс разработки пойдет тем быстрее, чем больше навыков имеет программист в конкретном языке. Под языками

программирования при сравнительном анализе также могут пониматься и системы визуальных скриптов, требующие лишь базовые знания в языке программирования или не требующие их вовсе. При выборе движка по языкам программирования разработчики опираются на собственные навыки и предпочтения.

Также на выбор среды разработки влияет возможность доступа к исходному коду среды. Доступ к исходному коду позволяет принять участие в ее доработке, использовать код для создания новых программ и исправления в них ошибок – через заимствование исходного кода, если это позволяет совместимость лицензий. Открытый исходный код позволяет разработчикам самостоятельно добавлять поддержку дополнительных устройств, а также оптимизировать или дополнить уже существующие. На данный критерий обращается внимание, только если в этом есть необходимость.

Определенно для игрового движка является преимуществом наличие официальной документации и обучающих материалов от его разработчиков. Этот критерий часто влияет на выбор среды начинающих разработчиков.

Для обзора были выбраны наиболее популярные среды разработки компьютерных игр с использованием технологий виртуальной реальности по версии независимой исследовательской компании Slant.co [8].

Самой популярной средой разработки по версии Slant.co является Unity 3d - кроссплатформенный игровой движок от компании Unity Technologies [9]. Unity представлен производителем в четырех версиях, цена которых варьируется от 0 $/мес до 125 $/мес. Бесплатной является только Personal-версия, предоставляющая ограниченный набор функционала и ограничивающая доступ к официальным обучающим материалам и исходному коду. Данная среда разработки имеет самую обширную поддержку платформ – более 25. Unity — самый популярный инструмент разработки среди XR-разработчиков, так как в ней разработано более 67% VR и AR приложений [9].

Следующий в рейтинге движок - Unreal engine 4 [10]. Это игровой движок, разрабатываемый и поддерживаемый компанией Epic Games. Unreal Engine является бесплатным начиная с 2015 года. Однако, если продукт, написанный на этом движке, начинает приносить разработчику прибыль (более 3000$ за квартал), лицензионным соглашением предусмотрены отчисления в пользу Epic Games в размере 5%. Особенностью данной среды является система визуального создания скриптов Blueprints Visual Scripting, основанная на использовании графического интерфейса. Такая система позволяет создавать и контролировать игровую логику без написания скриптов вручную. Количество поддерживаемых целевых платформ существенно меньше, чем у Unity, всего 13. На официальном сайте разработчиков представлена документация и обучающая информация. Имеется магазин с вспомогательными материалами для разработки — модели, скрипты, анимации, звуки, плагины и т.д.

В рейтинге также представлен App Game Kit VR [11] - это дополнение к среде App Game Kit Classic от компании The Game Creators, добавляет инструменты разработки VR-игр к основной версии движка. Следовательно, для запуска требуется Classic-версия движка, а она является платной ($49,99), собственно, как и само дополнение ($29,99). Среда поддерживает еще меньше целевых платформ, чем два предыдущих движка – всего 8. Однако преимуществом данной среды является возможность разработки одного единственного проекта для всех платформ.

Еще один движок LibGDX [12] – бесплатный игровой движок, разработанный компанией BadLogicGames. Он основан на языке программирования Java с некоторыми компонентами, написанными на языках C и C++, благодаря этому достигается высокая производительность движка. Как и App Game Kit, данная среда позволяет писать кроссплатформенные игры и приложения, используя один код. Поддерживает 7 целевых платформ: Windows, Linux, MacOS, Android, iOS, HTML5, Oculus Rift.

Следующий в списке CryEngine [13] – это коммерческий игровой движок, созданный немецкой компанией Crytek, предлагается для лицензирования другим компаниям. Движок ориентирован на разработку массовых многопользовательских онлайн игр. Скриптовая система среды базируется на языке Lua, который позволяет установку и настройку параметров игры, без использования кода C++. Особенностью движка является система игрового искусственного интеллекта, которая позволяет настроить реалистичное поведение неигровых персонажей, не касаясь кода C++. Целевых платформ, как и у LibGDX, 7 штук: Windows, iOS, Android, PS 4, Xbox One, Oculus Rift, HTC VIVE.

Результаты сравнения сред разработки по критериям, выявленным ранее, представлены в таблице 3.1.

Таблица 3.1 – Сравнение сред разработки по выделенным критериям

Среда разработки	**Цена**	**Поддерживаемая ОС**	**Языки программирования**	**Открытый исходный код**	**Доступ к обучающим материалам**	**Кол-во целевых платформ**
Unity 3D	От $0 до $125 в месяц	Windows, MacOS, Linux	C#	+ (только для Enterprise-версии)	Только у платных версий	25
Unreal engine 4	Бесплатно. Роялти Epic Games	Windows, MacOS, Linux	C++, Blueprints	+	+	13
AppGame Kit VR	$29,99 единоразово за дополнение, $49,99 единоразово за Classic-версию	Windows, MacOS, Linux	AGK Basic, C++, Pascal	-	-	8

LibGDX	Бесплатно.	Windows, MacOS, Linux	Java	+	-	7
CryEngine	Бесплатно. Роялти Crytec	Windows	C++, C#, Lua, Flowgraph	-	+	7

В результате сравнения было выявлено, что у каждой из рассмотренных сред разработки VR-игр есть как слабые стороны в отношении рассмотренных критериев, так и сильные. По результатам сбора и анализа материалов можно сделать вывод о том, что все рассмотренные среды разработки достойны внимания, однако выявить лучшую из представленных сред по всем критериям сразу не удается. Из этого можно сделать вывод, что движки нацелены на разные категории пользователей.

Если говорить о начинающих программистах и инди-разработчиках, создающих свои проекты без финансовой поддержки сторонних организаций, то тут речь надо вести о бесплатных движках с базовым функционалом, таких как LibGDX, и Personal-версия Unity 3d. Если опыта в программировании очень мало или нет совсем, то подойдут среды с системами визуальных скриптов и возможностью создания игр без написания кода. Как правило, начинающие разработчики редко преследуют прибыль в качестве главной цели создания проекта, чаще – получение и улучшение навыков разработки, следовательно, им также отлично подойдут движки Unreal engine 4 и CryEngine, установить которые также можно бесплатно.

Компании-разработчики игр, являющиеся коммерческими организациями, стремятся получить максимум прибыли с произведенных ими продуктов, поэтому при выборе движка они могут отказаться от Unreal engine 4 и CryEngine, так как в обоих случаях придется отчислять часть прибыли компаниям-разработчикам этих движков. Но проекты таких компаний обычно имеют значительную финансовую поддержку и для них могут быть использованы как платные версии сред разработки, так и бесплатные. Также коммерческие организации стремятся выбирать движки с

большим количеством целевых платформ, чтобы охватить как можно больше пользователей и, следовательно, получить больше прибыли.

Разработчикам, имеющим опыт создания компьютерных игр и нацеленных на получение впечатляющей графической реализации, следует обратить внимание на Unreal Engine 4.

Можно выделить среды разработки, одинаково подходящие всем группам пользователей – это Unity и Unreal Engine. В них сочетаются интуитивно понятный пользовательский интерфейс, простые панели инструментов, широкий выбор целевых платформ и языков для программирования сценариев, а также возможность бесплатной установки. Таким образом, были изучены и проанализированы различные среды разработки VR-игр.

Опираясь на результаты сравнительного анализа, были выделены ключевые критерии, повлиявшие на выбор среды разработки. Так как разработка ведется на операционной системе Windows, критерий «поддерживаемая ОС» не входит в набор ключевых критериев, влияющих на выбор среды разработки данной работы. Также на данном этапе критичного влияния на выбор не оказывают и наличие открытого исходного кода среды и количество целевых платформ. Поставленные задачи предполагается решить для ограниченного количества целевых платформ и без обращения к исходному коду среды для ее доработки. Однако, эти критерии могут перейти в набор ключевых при дальнейшей разработке и исследованиях, следовательно, наличие открытого исходного кода среды и большого количества целевых платформ при выборе движка будут его преимуществом.

Таким образом, главными критериями выбора среды разработки данной работы являются следующие: бесплатная лицензия, наличие доступа к официальным обучающим материалам и языки программирования, на которых ведется разработка. При выборе отдается предпочтение средам, поддерживающим C++, Java и любые системы визуального программирования. Данным критериям более всего соответствуют движки

Unreal Engine 4 и CryEngine. Unreal Engine 4 имеет преимущество перед вторым движком за счет открытого исходного кода и количества целевых платформ, в разы превышающего этот показатель у CryEngine. Так, для выполнения данного проекта в качестве среды разработки был выбран игровой движок Unreal Engine 4. В работе была использована самая последняя на момент разработки версия – Unreal Engine 4.24.3.

Так как игровой мир разработанной игры представлен в трехмерном пространстве, то объекты этого мира также в основном трехмерные. Для работы с 3D-ассетами был выбран 3DS MAX - это программное обеспечение для 3D-моделирования, анимации и визуализации. Был выбран именно этот редактор, так как на официальном сайте Unreal Engine опубликованы обучающие материалы с примерами интеграции в игровой движок созданных в данном редакторе объектов. При разработке с помощью данного продукта редактировались ассеты и создавалась анимация аватара. Этот 3D-редактор также имеет платную лицензию, но для студентов и преподавателей вузов предоставляется возможность бесплатного использования сроком на 3 года.

3.2. Функциональная спецификация проекта

Unreal Engine 4 предлагает разработку для нескольких платформ, поддерживающих BP: Google VR, Oculus VR, Samsung Gear VR, Steam VR, Windows Mixed Reality. Для реализации данного проекта была выбрана платформа Steam VR.

Steam VR – это среда выполнения в составе клиента Steam, которая обеспечивает работу приложений, поддерживающих BP. Выбор этой платформы обусловлен тем, что она поддерживает самый широкий круг устройств, среди всех перечисленных. Используя Steam VR, можно играть в VR-игры используя гарнитуру HTC Vive, Oculus Rift, Windows Mixed Reality и многие другие, поддерживающие Steam VR, список которых продолжает увеличиваться. Кроме того, данная среда позволяет использовать устройства,

не предназначенные для данной игры, она становится так называемым посредником между ПО и VR-устройством.

Таким образом, системные требования проекта будут определяться системными требованиями Steam VR. А также для развертывания приложения необходимо иметь VR-устройства, поддерживающие Steam VR.

Разрабатываемое игровое приложение представляет собой трехмерную ВР-версию игры «Сапёр» от первого лица. Приложение должно отвечать следующим требованиям:

1. До начала игры пользователю должны быть предложены уровни сложности на выбор.
2. Игровой процесс должен соответствовать правилам канонической версии игры, а именно:
    - поле игры должно состоять из ячеек квадратной формы;
    - игрок должен иметь возможность перемещаться по полю;
    - в начале игры все ячейки должны быть закрыты;
    - игрок знает количество мин, расположенных на поле;
    - игрок может открыть или пометить ячейку;
    - при открытии ячейки она отмечается числом, обозначающим количество мин, расположенных на окружающих ее 8-ми соседних ячейках;
    - при открытии ячейки с миной игра заканчивается – игрок проиграл;
    - при открытии всех ячеек, не содержащих мину, игра заканчивается – игрок победил.
3. Поле игры должно быть реализовано таким образом, чтобы игрок смог хорошо и быстро ориентироваться на нем.
4. Игрок должен иметь возможность поставить игру на паузу.
5. Игрок должен иметь возможность сменить настройки сложности.

6. Игра должна сохранить в себе основные особенности казуального жанра, такие как:
    - отсутствие долгого обучения игровому процессу;
    - простое и интуитивно-понятное управление.
7. Пользовательский интерфейс должен быть прост и понятен пользователю.
8. Игровой мир должен соответствовать тематике игры, помогать пользователю ориентироваться на уровне, не раздражать его и не мешать игровому процессу.

Для реализации проекта был использован ноутбук HP 15-ba055ur со следующими техническими характеристиками:

- операционная система: Windows 10 Home (64x);
- процессор: AMD A6 7310, 2000 МГц;
- объем оперативной памяти: 4Гб;
- видеокарта: AMD Radeon R5 M430.

3.3. Описание архитектуры программной системы

Система визуального программирования Blueprint является ключевым компонентом игрового движка Unreal Engine. Для создания новых элементов игрового процесса могут использоваться два метода – C++ и Blueprint. Работа с Blueprint-системой основывается на использовании нодов и проводов. Нод – визуальное представление функций, переменных, условий и операторов. Провода обеспечивают передачу данных между нодами. Визуальная среда программирования не требует использования традиционной текстовой IDE, но в то же время она гибкая и мощная, позволяет использовать практически весь потенциал программирования, полностью раскрывает функциональные возможности игрового движка. Блюпринт-скрипты довольно эффективны, так как их компиляция происходит на уровне байт-кода, а также процесс программирования исключает возникновение ошибок в синтаксисе

программного кода и не требует затрат времени на его оформление. Blueprint имеет графическую отладку, которая напоминает бегущий по проводам ток. А также разработчики Unreal Engine предоставляют большое количество официальной документации, обучающего материала и готовых обучающих проектов по работе с Blueprint-системой. Кроме того, создание классов на языке C++ не исключает необходимости работы в той или иной степени с системой Blueprint. Опираясь на данные факты, в качестве способа программирования для данной работы была выбрана система визуального программирования.

Структурной единицей Blueprint-системы является блюпринт. В Unreal engine существует пять типов блюпринтов: блюпринт уровня, блюпринт-класс, data-only-блюпринт, блюпринт-макрос, блюпринт-интерфейс. Наиболее часто используемые типы - это блюпринты уровня и блюпринт-классы.

Под архитектурой программной системы в данной работе понимается основная структурная организация блюпринтов этой системы. При проектировании архитектуры было выделено несколько ключевых компонентов, которые будут описаны далее.

MainGameMode блюпринт-класс - класс игрового процесса. Он наследуется от родительского класса GameMode. Используется для задания правил игры, отслеживает ход игры, управляет ее функциональностью.

VRPawn - блюпринт-класс аватара, представляет аватара в игре и управляет им, наследуется от родительского класса Pawn.

VRController – блюпринт-класс контроллера, содержит визуальное представление и функционал устройства-контроллера. Он наследуется от родительского класса Actor.

Совокупность блюпринтов, управляющих правилами игры, аватарами, камерой и виджетами в проекте, составляет систему Gameplay Framework. Блюпринт-классы GameMode, Actor и Pawn составляют часть базового Gameplay Framework проекта Unreal Engine.

Cell – блюпринт-класс ячейки поля. Он содержит визуальное представление и функционал одной ячейки игрового поля.

Блюпринты уровней управляют событиями, происходящими на конкретном уровне. Под уровнем понимается ограниченное физическое пространство, на котором происходит действие. В данном проекте используются два типа уровня: игровые уровни и уровень-интерфейс (рисунок 3.2). Весь игровой процесс происходит в рамках трех игровых уровней, соответствующих трем размерам и дизайнам игрового поля. Весь пользовательский интерфейс происходит на отдельном уровне, так как он представлен также в пространстве BP.

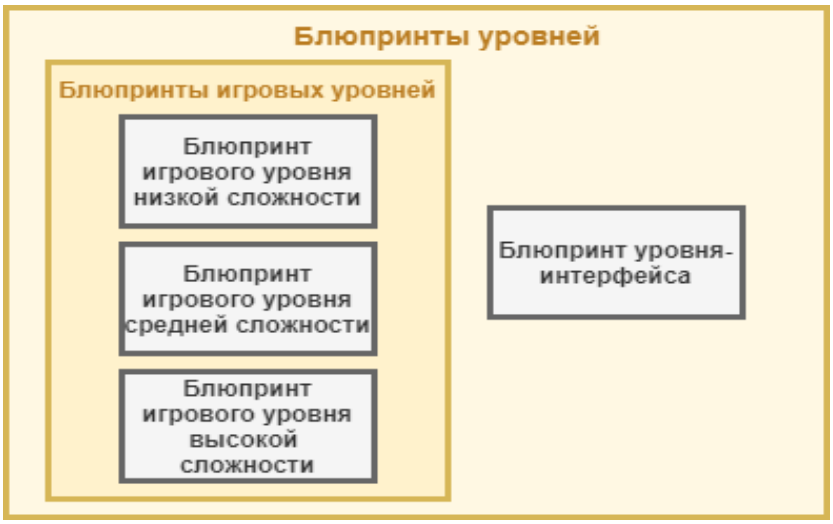

Рисунок 3.2 – Классификация блюпринтов уровней проекта

Блюпринты виджетов содержат виджеты, которые обеспечивают интерфейс игры. Классификация виджетов представлена на рисунке 3.3.

Рисунок 3.3 - Классификация блюпринтов виджетов проекта

Между выделенными компонентами имеется зависимость «многие ко многим», в связи с чем появляется неудобство при обращении компонентов друг к другу, приходится заводить в каждом из них поля для хранения ссылок. Следовательно, в данной ситуации целесообразным является использование архитектурного шаблона «Посредник». Компонент-посредник выступает в качестве центра связи между остальными компонентами, избавляя от необходимости компонентов ссылаться друг на друга, а связь «многие ко многим» заменяется связью «один ко многим». В качестве компонента-посредника выступает блюпринт-класс MainGameInstance. Данная архитектурная система представлена в формате блок-схемы на рисунке 3.4.

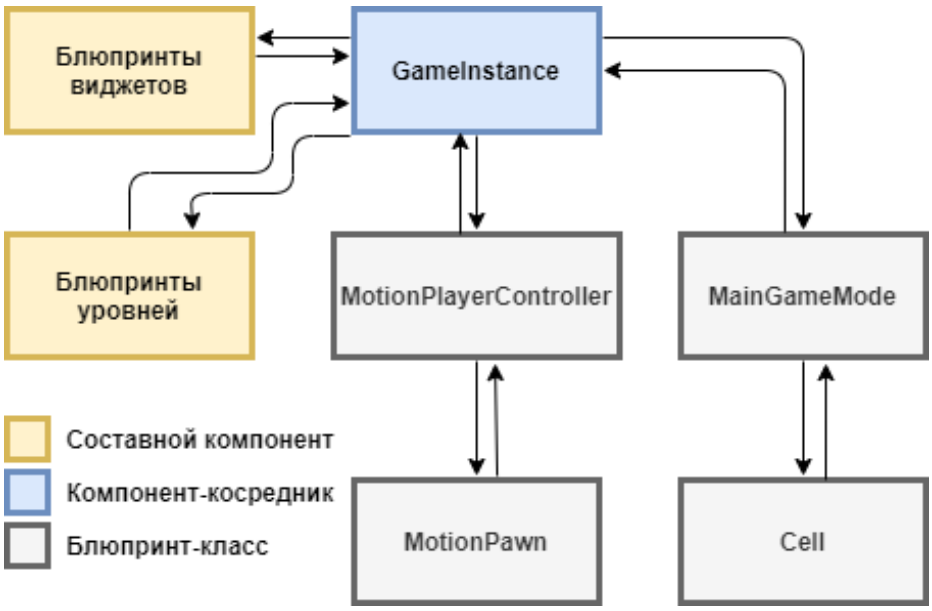

Рисунок 3.4 - Архитектура программной системы

MainGameInstance хранит ссылки на другие компоненты, а также глобальные переменные, которые задают настройки игры или состояние процесса игры. Также данный класс отвечает за организацию перемещения по уровням. В связи с посредником не нуждается компонент Pawn, так как он связан только со своим контроллером и лишь выполняет его команды, а также класс Cell, так как экземпляры этого класса создаются и используются только классом MainGameMode.

3.4. Функциональное проектирование UI

Пользовательский интерфейс игры является частью игровой механики, он предоставляет возможность пользователю взаимодействовать с игрой. Основную часть UI в настоящем проекте составляют пользовательские меню: главное меню, меню паузы и другие. UI также нуждается в грамотном проектировании, так как впервые открыв игру, пользователь сталкивается именно с ним. Так, плохо спроектированный интерфейс может испортить впечатление от игры с первых минут.

Были определены основные принципы в проектировании интерфейса в рамках данного проекта:

- простота и интуитивная понятность при взаимодействии;
- минимальность требуемых ресурсов от пользователя;
- максимальное взаимодействие пользователя с интерфейсом.

Простота и интуитивная понятность достигается за счет группировки элементов интерфейса по тематике функциональности, следовательно, единовременно пользователь видит на дисплее меньше элементов и не теряется в них. Выводимая информация не должна требовать интерпретации, должна быть легко читаемой и легко различаемой.

Минимальность требуемых ресурсов подразумевает отсутствие сложных операций ввода, исключение повторяющихся действий.

Максимальное взаимодействие означает то, что интерфейс должен поддерживать пользователя, он не должен отвлекаться на поиск информации.

Руководствуясь описанными выше принципами, были разработаны сценарии использования для меню игры на языке UML, представленные на рисунке 3.5.

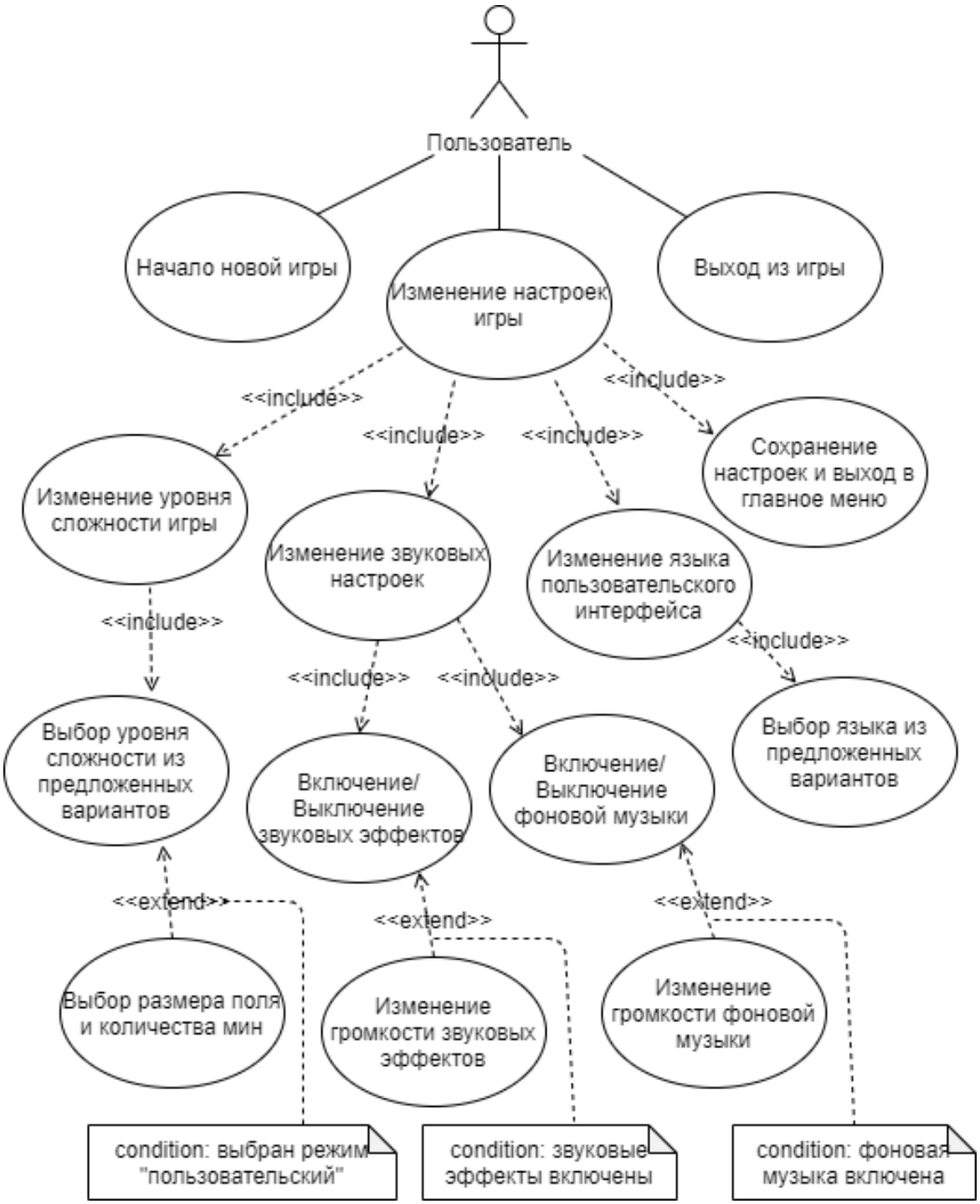

Рисунок 3.5 – UML диаграмма сценариев использования игрового меню

Также были разработаны сценарии использования для меню паузы, UML диаграмма которых представлена на рисунке 3.6. В данном случае из настроек были оставлены только звуковые настройки, так как они могут понадобиться в процессе игры.

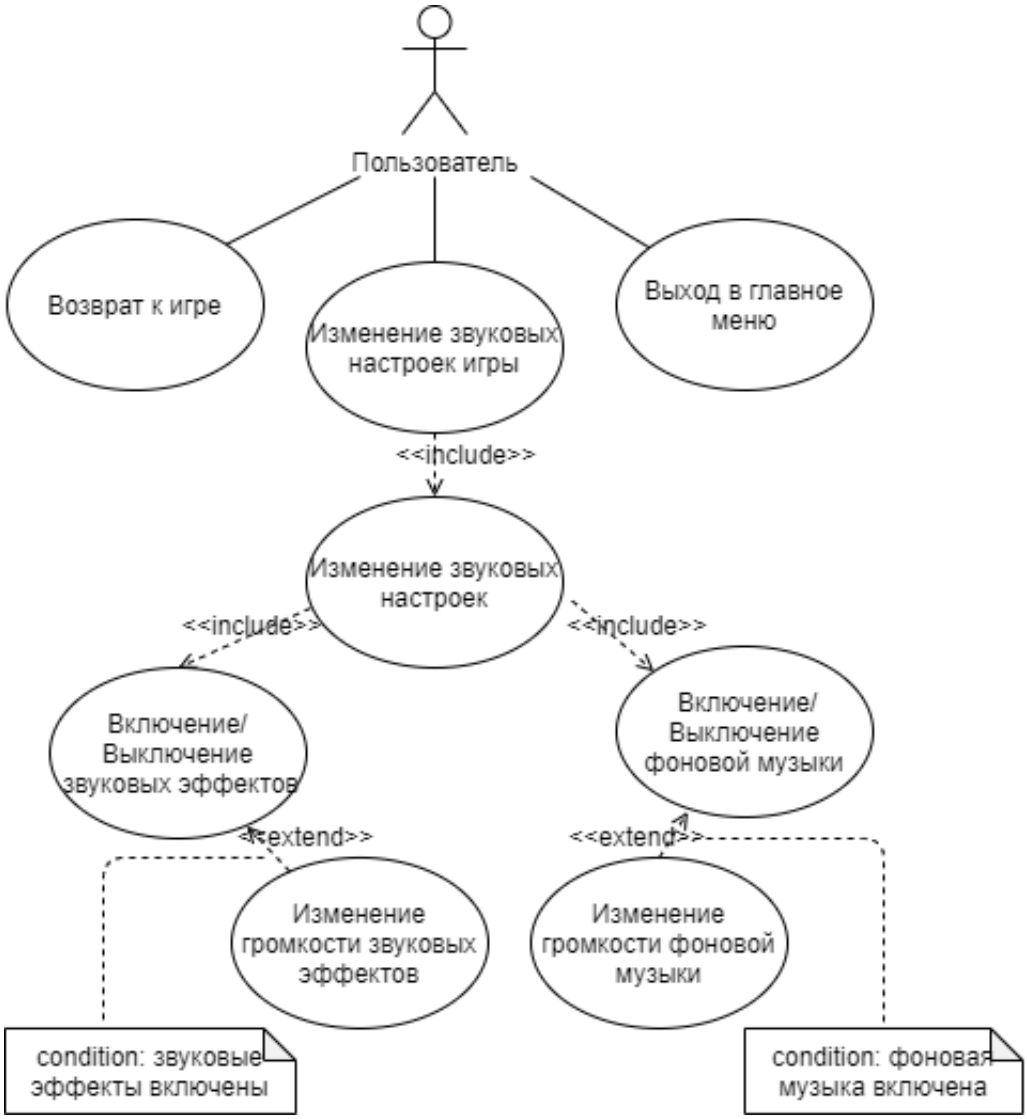

Рисунок 3.6 – UML диаграмма сценариев использования меню паузы

Разработанные сценарии использования удовлетворяют принципам, описанным ранее, а также требованиям к пользовательскому интерфейсу, описанных в подразделе 2 настоящего раздела.

3.5. Эскизное проектирование внешнего представления игрового процесса и UI

В рамках проектирования внешнего вида игрового процесса в графическом редакторе Adobe Photoshop был разработан эскиз, который приведен на рисунке 3.7.

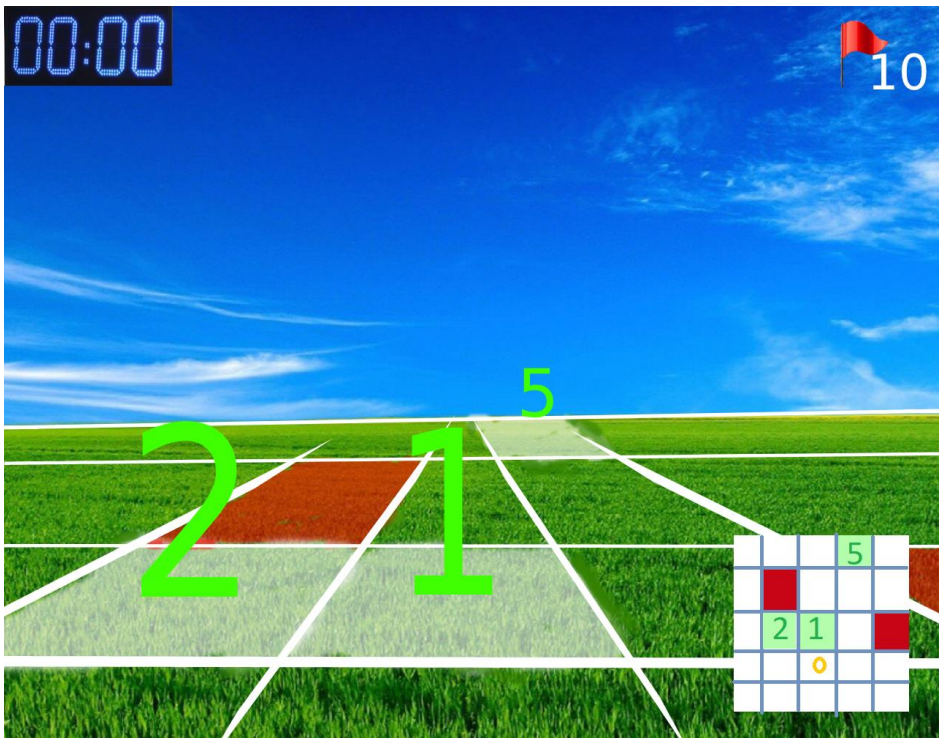

Рисунок 3.7 – Эскиз внешнего вида игрового процесса

В результате проектирования данного эскиза была решена задача переноса двумерного игрового поля в трехмерное пространство ВР. Игровое поле располагается на местности, по которой может перемещаться пользователь. Разработанный эскиз решает большую часть функциональных требований, описанных в подразделе 2 данного раздела. При его создании учитывалось, что пользователь должен находиться на игровом поле, состоящем из ячеек и иметь возможность по нему перемещаться. Он должен видеть в поле зрения достаточное количество ячеек, чтобы ориентироваться в игре. Игрок может пометить или открыть ячейку, он должен чётко видеть и различать помеченные ячейки и цифры на открытых ячейках. Также пользователь должен иметь возможность видеть количество оставшихся флагов, которые он может поставить, таймер игры и карту в реальном времени в привычной для игры «Сапёр» проекции «сверху», чтобы видеть все окружающие его ячейки.

В рамках проектирования внешнего представления UI был разработан эскиз, который приведен на рисунке 3.8.

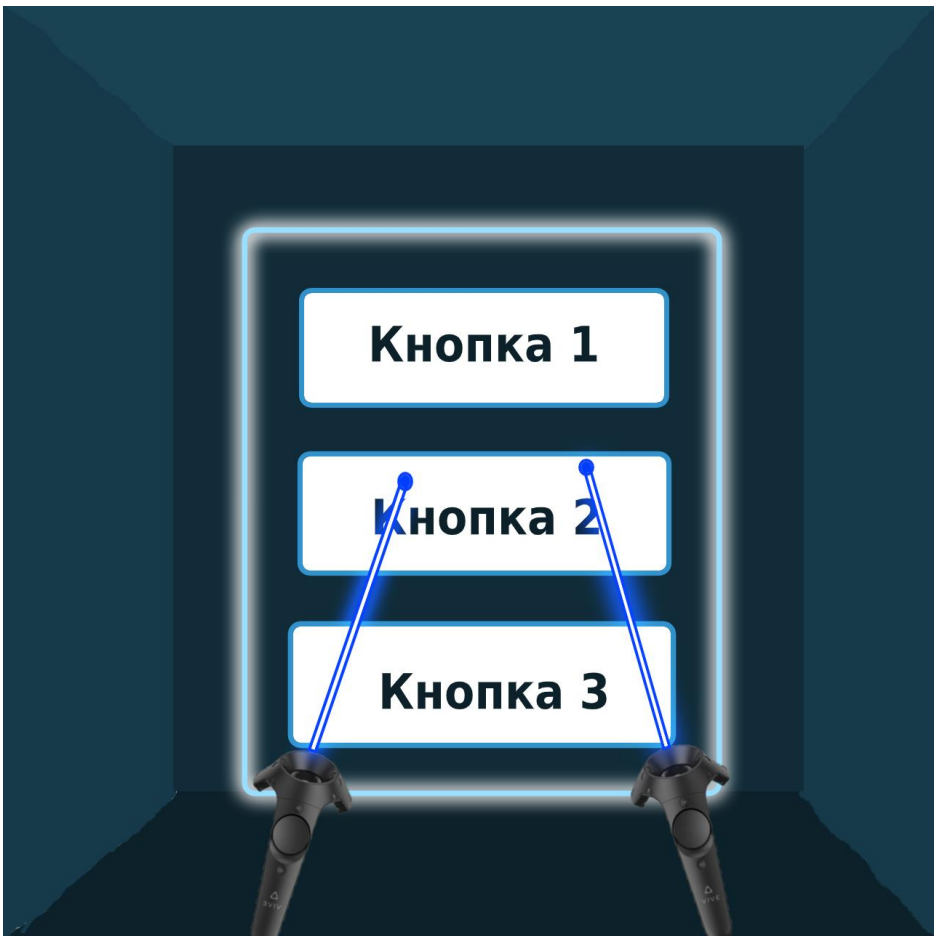

Рисунок 3.8 – Эскиз внешнего представления UI

Разработанный эскиз также удовлетворяет требованиям, поставленным к пользовательскому интерфейсу. Пользователь находится в трехмерном пространстве, следовательно, не снижается уровень погружения. При этом пользователь хорошо видит и различает виджет и может управлять лучом взаимодействия как курсором на мониторе. Такое управление является простым и интуитивно понятным для него.

Выводы

В данном разделе были рассмотрены и проанализированы различные среды разработки BP-игр. В результате сравнительного анализа сред была выбрана наиболее подходящая для реализации данного проекта – Unreal Engine 4. Кроме этого, были определены вспомогательные средства разработки, определены функциональные требования к проекту и выбрана целевая платформа – Steam VR. Также было разработано и описано решение

по структурной организации программной системы с выделением ключевых компонентов. В рамках проектирования системы пользовательского интерфейса были описаны сценарии ее использования, и разработан эскиз внешнего представления работы пользователя с UI. Также был разработан эскиз внешнего представления игрового процесса в соответствии с описанными функциональными требованиями.

Таким образом, в данном разделе ВКР были описаны ключевые этапы проектирования игрового приложения, что является подготовительной работой к дальнейшей реализации программного продукта.

4. РЕАЛИЗАЦИЯ ПО

4.1. Описание реализации игрового процесса

4.1.1. Описание экземпляра ячейки игрового поля

Игровое поле состоит из ячеек, то есть из экземпляров класса Cell (рисунки 4.1-4.2).

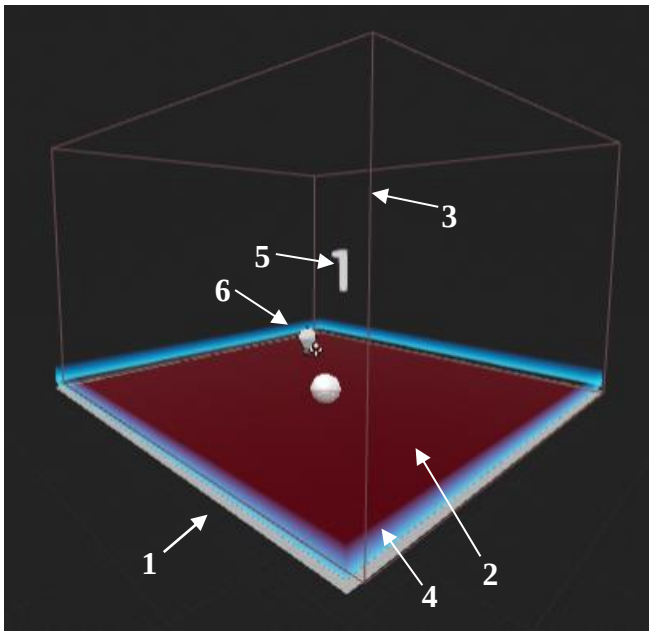

Рисунок 4.1 – Визуальное представление блюпринт-класса Cell с нумерацией компонентов

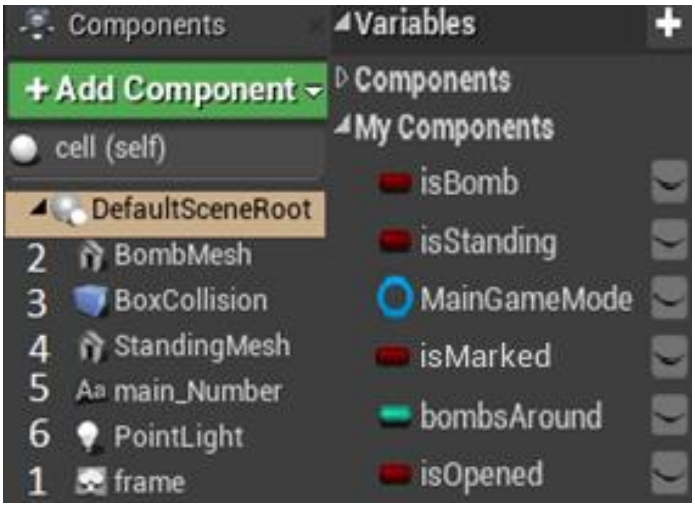

Рисунок 4.2 – Компоненты и переменные блюпринт-класса Cell

Ячейка состоит из компонентов, описание которых приведено в таблице 4.1.

Таблица 4.1 – Компоненты ячейки поля и их назначение

Обозначение на рисунке	Название компонента	Описание компонента
1	frame	Спрайт, обозначающий границы ячейки, который обеспечивает внешнее разделение игрового поля на сегменты-ячейки
2	BombMesh	Плоскость, окрашенная в красный полупрозрачный материал, которая становится видимой при установке флага на ячейку.
3	BoxCollision	Компонент коллизий, генерирует коллизии ячейки, необходим для обработки событий при нахождении игрока на конкретной ячейке.
4	StandingMesh	Компонент, подсвечивающий ячейку с помощью специального материала, используется для обозначения текущей ячейки, на которой находится игрок, за счет изменения параметров видимости.
5	main_number	Текстовый компонент, который отображает значение переменной bombsAround, становится видимым после открытия ячейки.
6	PointLight	Компонент точечного света, используется для подсвечивания компонента main_number.

Кроме компонентов важными функциональными элементами ячейки являются ее параметры, а именно переменные блюпринта Cell (таблица 4.2).

Таблица 4.2 – Параметры ячейки поля и их назначение

Название	Тип данных	Назначение
isBomb	boolean	Отображает наличие мины в ячейке. Если значение равно true, то ячейка «заминирована».
isStanding	boolean	Используется для отображения текущей ячейки, на которой находится игрок. Единовременно значение true может быть только у одной ячейки, на которой находится игрок.
isMarked	boolean	Отображает, была ли ячейка помечена игроком как «заминированная».
isOpened	boolean	Отображает состояние ячейки, если true, то ячейка открыта, иначе – ячейка еще не открывалась. Ячейка может быть открыта только один раз за игру.
bombsAround	integer	Отображает количество «заминированных» ячеек среди соседних с текущей.
GameInstance	MainGameInstance	Содержит ссылку на блюпринт-посредник MainGameInstance.

Ячейка является структурной единицей всего игрового поля. Визуализация игрового процесса реализуется в основном за счет изменения параметров компонентов ячейки, например параметров видимости и параметров отображения. Реализация игрового процесса основывается на чтении и изменении параметров ячеек поля.

4.1.2. Алгоритмы и принципы реализации игровой логики

Ключевым элементом игровой логики является поле игры. С каждым началом новой игры создается новое игровое поле. Общий алгоритм создания игрового поля представлен на рисунке 4.3.

Рисунок 4.3 – Общий алгоритм создания игрового поля

Поле игры, состоящее из ячеек генерируется в начале каждой новой игры. Генерация происходит за счет спаунинга из блюпринт-класса MainGameMode акторов блюпринт-класса Cell на уровень, то есть поле игры состоит из экземпляров класса Cell. Под спаунингом понимается процесс создания нового экземпляра актера с определенными преобразованиями. Спаунинг используется тогда, когда помещать отдельные актеры вручную на уровень становится нецелесообразным. В данной разработке требуется поместить на уровень большое количество одинаковых актеров с разным местоположением с возможностью в дальнейшем к ним обращаться, в качестве решения применяется спаунинг актеров класса Cell.

Генерация поля начинается с нижнего левого угла поля и основывается на двойном цикле, в каждой итерации которого спаунится одна ячейка и помещается в массив CellArr (массив ячеек поля). Поле игры целесообразнее

всего хранить в двумерном массиве, однако в движке Unreal Engine отсутствует возможность создания многомерных массивов и работы с ними. Поэтому CellArr представляет собой одномерный массив ссылок на экземляры класса Cell. Схема алгоритма генерации ячеек представлена на рисунке 4.4. При генерации новой ячейки, она сдвигается на величину стороны ячейки (CellSize) по оси X и/или по оси Y. Размеры игрового поля определяются значением n=FieldSize-1, где значение FieldSize соответствует количеству рядов (столбцов) ячеек, содержащихся в каждой стороне поля.

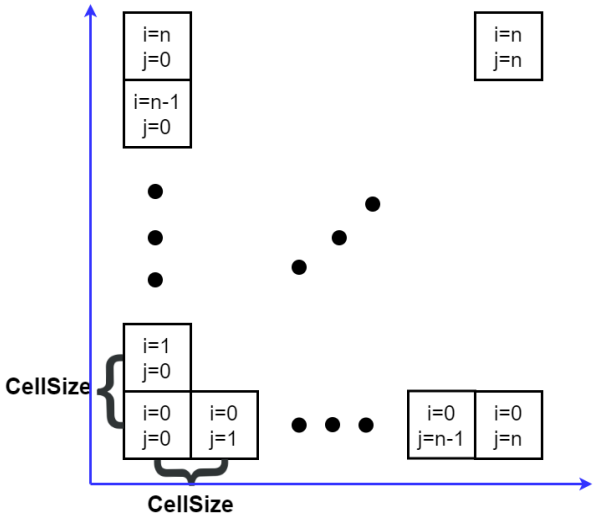

Рисунок 4.4 - Генерация ячеек игрового поля

Далее нужно расставить мины на ячейки, для этого необходимо сгенерировать рандомным образом местоположения каждой мины. Это происходит в методе GenerationBobmsInd класса MainGameMode (рисунок 4.5). С помощью метода RandomIntegerInRange генерируются индексы массива CellArr, т.е. ячейки, которые будут содержать мину, и помещаются в массив IndBombs. Количество мин определяется переменной MainFLC, которая задается в настройках игры. После этого для каждой ячейки, соответствующей сгенерированному индексу массива CellArr вызывается метод SetBomb, который изменяет значение поля isBomb типа boolean на true.

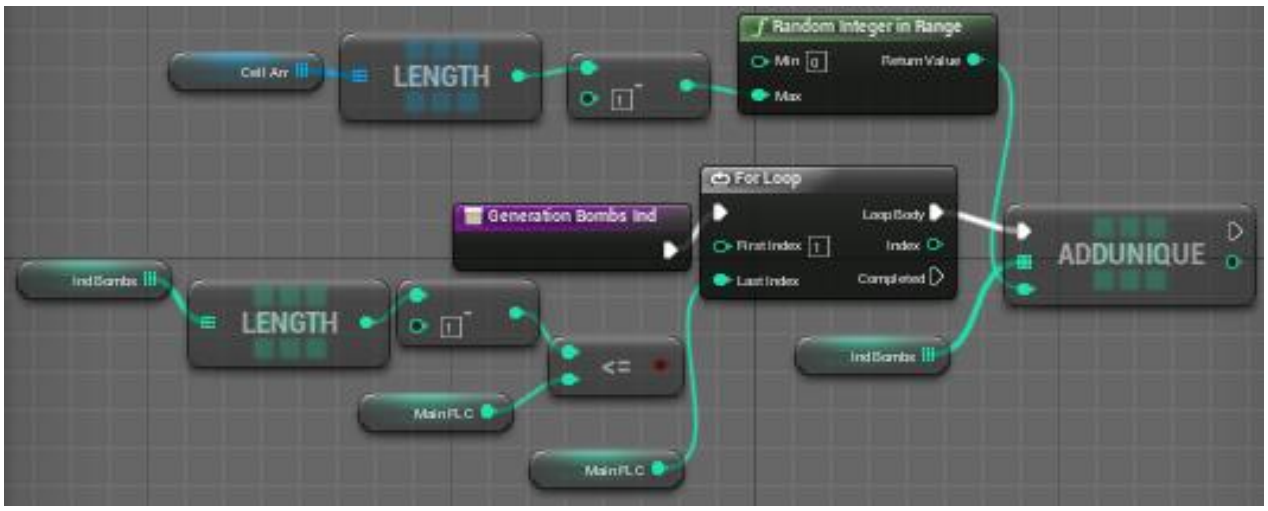

Рисунок 4.5 - Генерация номеров ячеек, которые содержат мины

После того как мины установлены, необходимо рассчитать значение параметра BombsAround, для каждой ячейки, не содержащей мину. Для этого происходит обращение ко всем 8 смежным с данной ячейкам по индексам массива CellArr и проверка значения их параметра isBomb. В качестве значения BombsAround выступает количество истинных значений параметра isBomb смежных ячеек. Индексы смежных ячеек для ячейки с индексом i представлены на рисунке 4.6. Для корректной работы алгоритма, в частности для ячеек, расположенных на границах поля, необходимы дополнительные условия, проверяющие валидность этих ячеек.

CellArr[i+ FieldSize -1]	*CellArr[i+ FieldSize]*	*CellArr[i+ FieldSize +1]*
CellArr[i-1]	***CellArr[i]***	*CellArr[i+1]*
CellArr[i- FieldSize -1]	*CellArr[i- FieldSize]*	*CellArr[i- FieldSize +1]*

Рисунок 4.6 – Вычисление индексов смежных ячеек

Обработка событий ввода происходит в блюпринте VRPawn. В рамках игрового процесса пользователь может открыть или пометить ячейку. Когда

пользователь стоит на ячейке, он вызывает событие перекрытия для коллизии этой ячейки (рисунок 4.7). Тогда события ввода выполняются для текущей ячейки, на которой находится аватар.

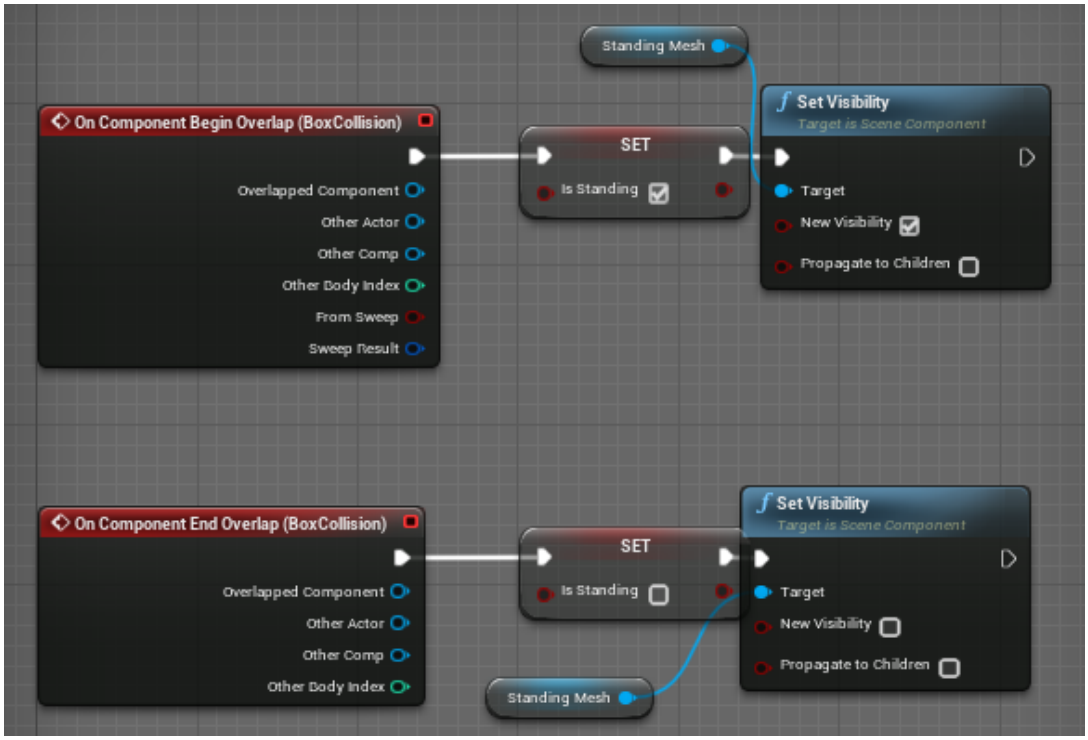

Рисунок 4.7 – Обработка события перекрытия коллизий ячеек

Событие открытия ячейки вызывает (через блюпринт-посредник) метод mainCellOpening блюпринта MainGameMode. Этот метод делает видимым компонент main_Number ячейки и изменяет переменную isOpened на значение true. Если поле BombsAround было равно 0, то есть у данной ячейки нет соседей, содержащих мину, тогда метод mainCellOpening рекурсивно вызывается для каждой из смежных с данной ячеек. Таким образом открывается вся область пустых ячеек (рисунок 4.8).

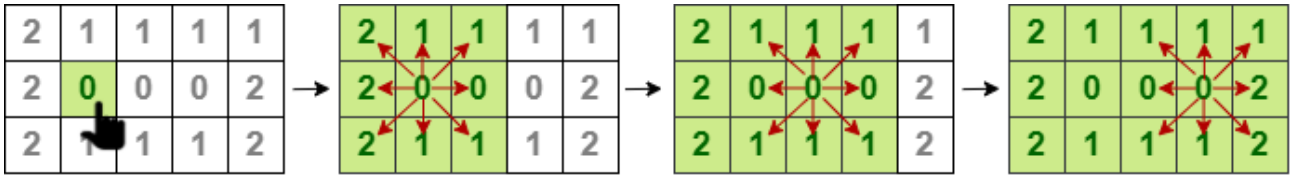

Рисунок 4.8 – Пример открытия области ячеек

Событие метки ячейки вызывает метод Marking класса Cell, которое изменяет видимость компонента BombMesh и изменяет значение переменной isBomb.

4.2. Организация пользовательского взаимодействия и перемещения в пространстве ВР

Вся логика пользовательского взаимодействия и перемещения в пространстве ВР описывается в блюпринтах VRPawn и VRController. VRPawn блюпринт состоит из компонентов (рисунок 4.9):

- основная камера (CameraComponent), через которую пользователь смотрит на происходящее;
- компоненты, необходимые для реализации отображения дополнительной карты (SpringArm, minimapCapture, PlayerSprite);
- виджет, отображающий всплывающие перед пользователем дополнительную карту и меню паузы (WidgetComponent).

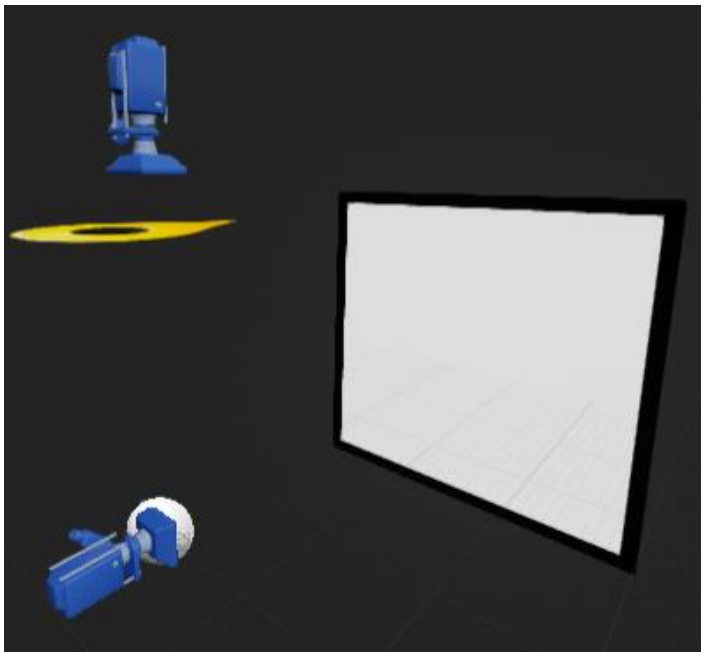

Рисунок 4.9 – Визуальное представление блюпринт-класса VRPawn

Главные функции VRPawn блюпринта – это обработка событий ввода, совершаемых пользователем и организация вида от первого лица. Актер VRPawn помещается на уровень и соответствует аватару в игре. После

начала игры в VRPawn спаунится два актера VRController, соотвующие правому и левому контроллеру (рисунок 4.10). В VRController блюпринте описывается основная реализация возможностей аватара.

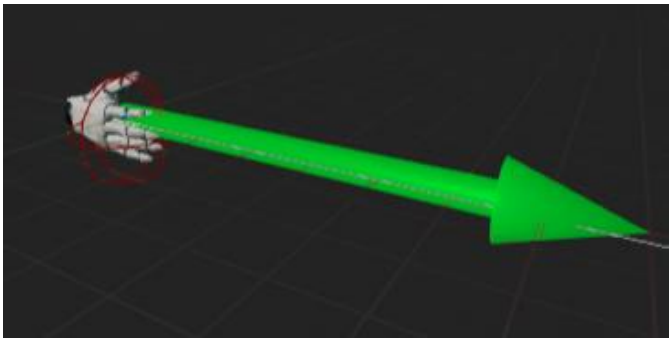

Рисунок 4.10 - Визуальное представление блюпринт-класса VRController

Взаимодействие пользователя с интерфейсом игры реализуется с помощью компонента MainInteraction, представляющего из себя луч, исходящий из контроллера. Это экземпляр компонента WidgetInteractionComponent, который создан специально для взаимодействия с виджетами. Он позволяет по принципу курсора направлять фокус взаимодействия туда, куда направлен контроллер. Виджеты реагируют на нажатие левой кнопки компьютерной мыши. Это событие было расширено, используя методы PressPointerKey и ReleasePointerKey. В результате события Press и Release, вызываемые нажатием определенных кнопок на контроллере, сигнализируют виджетам о специальных действиях, соответствующим нажатию и отпусканию левой кнопки мыши.

Как уже было сказано, класс VRPawn унаследован от родительского класса DefaultPawn. В классе DefaultPawn уже реализовано стандартное управление аватаром, которое включено по умолчанию, в которое входят: движение вперед, назад, влево, вправо и прыжок. В целом, такой тип перемещения удовлетворяет поставленным требованиям к разработанному приложению. Однако, на практике перемещение по полю таким образом сильно замедляет ход игры, ведь в двумерной версии у пользователя есть возможность взаимодействовать со всеми ячейками поля в любом порядке. А

если пользователю необходимо взаимодействовать последовательно с двумя ячейками, находящимися на противоположных концах поля, то ему для этого придется пересечь все поле в пространстве ВР описанным образом, чтобы добраться до второй ячейки - это сильно увеличит время игры. Чтобы решить эту проблему, было решено добавить альтернативный способ перемещения по полю игры – телепортацию. Телепортация часто используется в играх виртуальной реальности в качестве основного или дополнительного способа передвижения аватара.

Для реализации телепортации в VRController было добавлено еще несколько компонентов (рисунок 4.11): ArcDirection – компонент-стрелка, определяющая направление луча телепортации; ArcSpline – сплайн, определяющий положение луча телепортации; ArcEndPoint – сфера, центр которой определяет точку-местоположение телепортации, находится в конце луча телепортации; компоненты, визуализирующие целевое местоположение телепортации и ее направление.

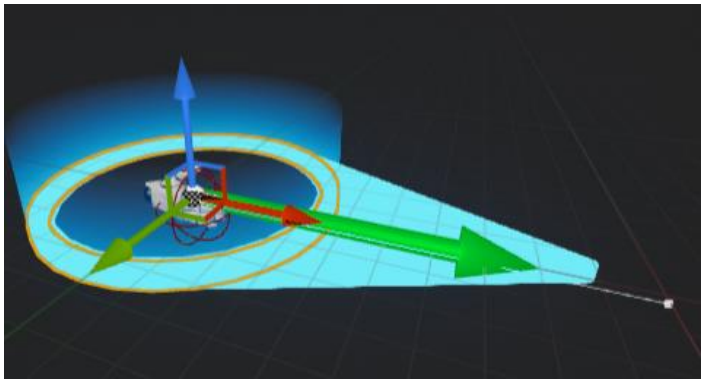

Рисунок 4.11 - Визуальное представление блюпринт-класса VRController после добавления компонентов для телепортации

При нажатии клавиши или сочетания клавиш, определенных для события телепортации, в VRPawn вызывается метод из VRController, активирующий телепортацию, который выстраивает луч телепортации и тем самым определяет местоположение и направление будущей телепортации. При отпускании данных клавиш, вызывается метод текущего класса,

выполняющий телепортацию в это местоположение, и устанавливает аватара в соответствии с выбранным направлением (рисунок 4.12).

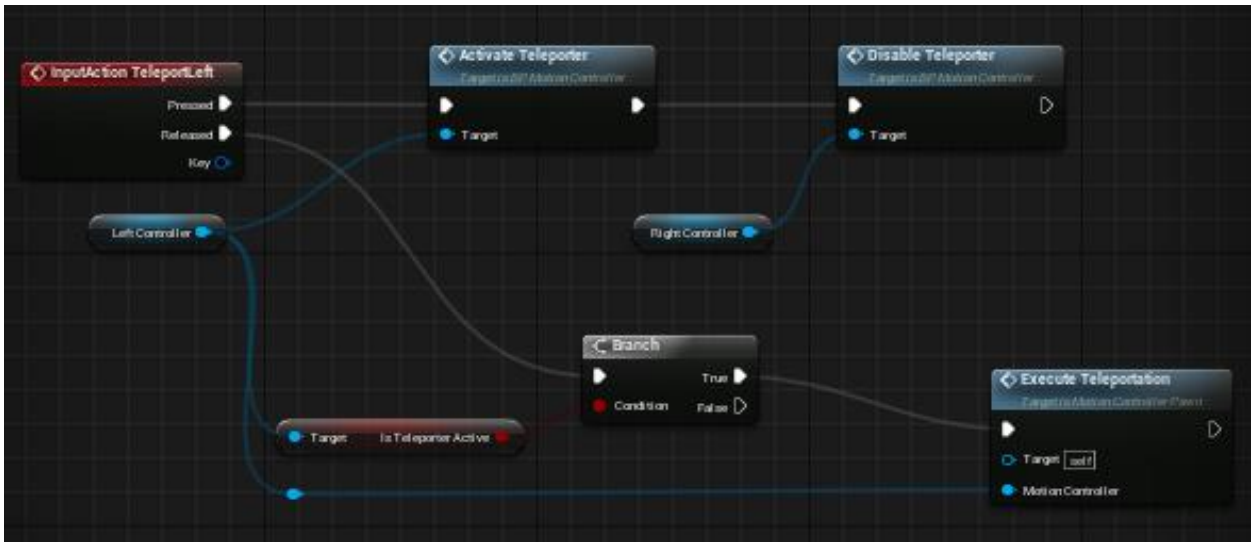

Рисунок 4.12 – Обработка события телепортации для левого контроллера

Луч телепортации представляет собой траекторию летящего виртуального объекта (рисунок 4.13).

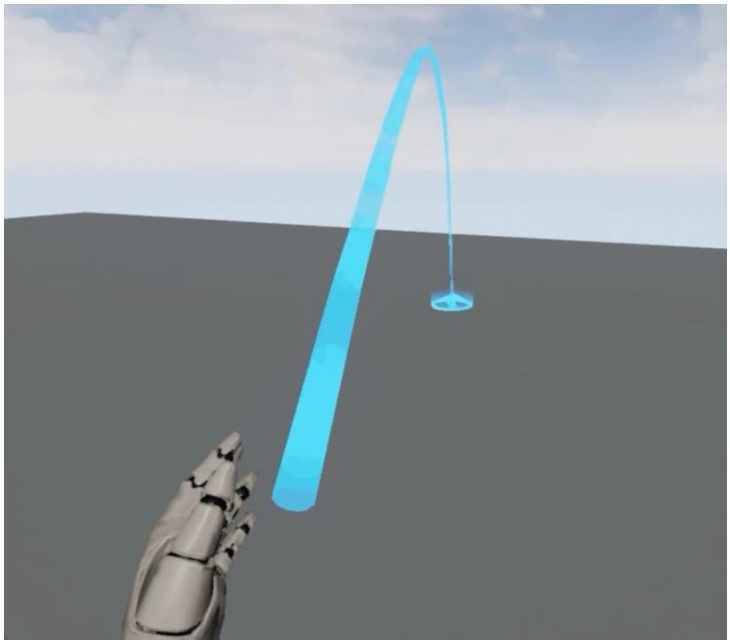

Рисунок 4.13 – Луч телепортации

Начало луча находится в контроллере, а его конец указывает на место будущей телепортации. Траектория летящего объекта строится с помощью специального метода PredictProjectilePathByObjectType, который предсказывает траекторию объекта, подчиняющегося законам физики и

который был брошен со стартовой позиции с некоторой векторной скоростью.

При выполнении телепортации изменяется местоположение аватара в пространстве с помощью метода Teleport.

Такой метод перемещения в пространстве ВР является удобным и быстрым, не требующим лишних действий от пользователя и не отвлекающим его от процесса игры.

4.3. Описание решений по созданию игрового мира

Создание мира – это процесс помещения актеров и художественных ассетов на уровень [14]. Хорошо проработанный игровой мир вызывает желаемый эмоциональный отклик у игроков. С помощью него можно направить и помочь игроку ориентироваться на уровне, создать визуальное повествование и повысить уровень погружения в игровой процесс.

При разработке дизайна игровых уровней в данной работе, преследовалась цель не создать полного расхождения с военной тематикой игры, но максимально снизить ее агрессивность.

Главной целью в создании мира ВР является его реалистичность. Чем выше реалистичность, тем быстрее и глубже пользователь погружается в виртуальный мир и его захватывает игровой процесс.

Создание мира происходит в несколько этапов.

Определение границ. Границы игрового мира играют большую роль в его создании: слишком узкие границы снизят уровень погружения, а слишком широкие будут задействовать излишние ресурсы. Границы передвижения аватара в данном приложении определяются границами игрового поля, далее он не может передвигаться. Следовательно, границы мира будут определены тем, как далеко и что именно может видеть аватар.

Определение масштаба. Не менее важным этапом является определение масштабов игрового мира и объектов в нем находящихся. Выбранный для них масштаб должен быть максимально реалистичен, чтобы

игрок не чувствовал себя слишком маленьким или слишком большим в мире, а объекты не казались искаженными и ненатуральными.

Создание наброска мира. Этот этап подразумевает добавление глобальных объектов на уровень, таких как водоемы, горы, стены, здания и т.д. Этот этап создает архитектурный каркас для уровня.

Декорация мира. На этом этапе создается внешнее оформление уровня, максимально прорабатывается его реалистичность, этап подразумевает добавление растительности, камней, предметов интерьера и т.д. Также на этом этапе могут быть добавлены световые и аудио-компоненты, которые создают визуальное впечатление и настроение.

Для проработки игрового мира в данной работе использовался бесплатный контент, предлагаемый Epic Games.

4.4. Организация тестирования

В процессе разработки и по её итогам проводилось тестирование разработанного приложения. Для тестирования VR-приложения может быть использовано два подхода:

1. тестирование с помощью VR-оборудования
2. эмуляция VR-оборудования

В рамках данной работы был использован второй подход. Эмуляция VR-устройств была выполнена с помощью специализированного ПО под названием Riftcat. Подключение к движку производилось с помощью SteamVR.

Riftcat позволяет эмулировать наличие высокотехнологичных VR-девайсов с помощью обычного смартфона. Это возможно благодаря тому, что современные смартфоны оснащены гироскопом и другими датчиками положения устройства в пространстве. Для этого необходимо установить ПО Riftcat на компьютер и мобильное приложение VRidge на смартфон. После подключения смартфона по сети Wi-Fi, на экран смартфона выводится изображение в таком виде, в каком оно выводилось бы на дисплей VR-

шлема, но с одним исключением. На смартфоне экран разделен на 2 части, изображение выводится для левого и для правого глаза по принципу стереоскопического зрения (рисунок 4.14). Для комфортного тестирования были использованы очки виртуальной реальности для смартфона VR BOX, выполняющие только функцию стереоскопа.

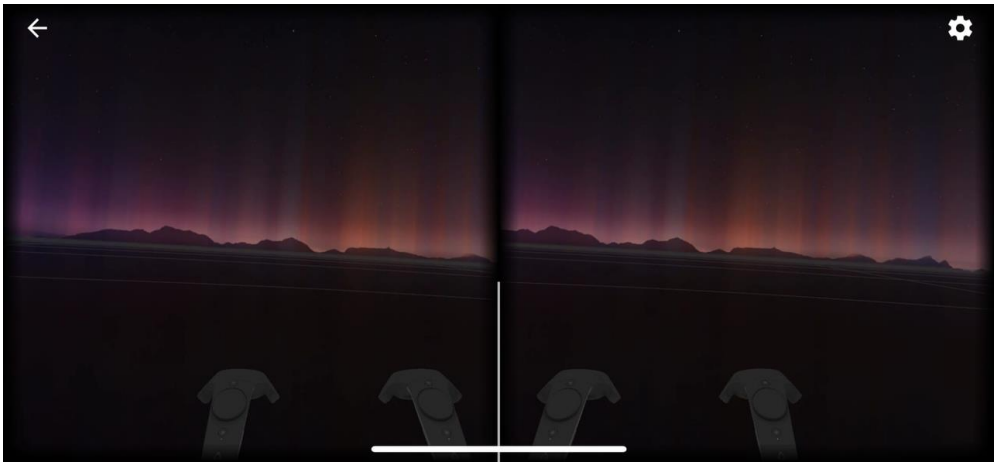

Рисунок 4.14 – Экран смартфона в приложении VRidge

Для эмуляции контроллеров также был использован Riftcat. Он позволяет имитировать действие контроллеров с помощью компьютерной мыши и клавиатуры. Riftcat позволяет самостоятельно устанавливать сочетания клавиш клавиатуры, которые будут соответствовать кнопкам на контроллере. Таким же образом можно изменять положение виртуальных контроллеров в пространстве.

На практике такой способ тестирования оказался довольно удобен и экономичен. Однако из-за подключения смартфона по Wi-Fi, могут возникать небольшие задержки. Имитированный таким образом VR-шлем выполняет все функции полноценного. Имитированные контроллеры выполняют свою функцию в рамках данного проекта, так как их использование в приложении ограничено (акцент сделан на нажатие кнопок, а не на управление). Однако в рамках другого проекта, в котором может потребоваться активное их перемещение в пространстве, такой способ может оказаться бесполезным.

Выводы

В данном разделе были описаны ключевые решения по разработке настоящего игрового приложения.

В рамках описания реализации игрового процесса был представлен принцип создания и организации игрового поля – спаунинг экземпляров класса ячейки на уровень, а также приведено подробное описание класса ячейки. В данном разделе также представлена структурная и функциональная организация классов VRPawn и VRController, описаны принципы взаимодействия пользователя с UI посредством VR-устройств. В процессе реализации перемещения игрока в пространстве виртуальной реальности выяснилось, что стандартного перемещения аватара по полю недостаточно для комфортной и быстрой игры. Так, был добавлен дополнительный способ перемещения – телепортация, подробное описание реализации которой также приведено в текущем разделе. Также в разделе было описано поэтапное создание игрового мира, и приведены решения по организации тестирования VR-приложения с помощью ПО Riftcat и стерео-очков для смартфона.

5. АНАЛИЗ РЕЗУЛЬТАТОВ

5.1. Описание результатов разработки

Результатом проделанной работы является игровое VR-приложение. Для его полноценного использования необходимо иметь VR-гарнитуру – HMD и контроллеры. Подключение этой гарнитуры реализуется через платформу SteamVR.

При запуске приложения пользователь попадает в главное меню, из которого он может начать игру, изменить настройки будущей игры или покинуть игру (рисунок 5.1).

Рисунок 5.1 – Главное меню игры

Перед началом новой игры рекомендуется проверить и изменить, если это требуется, настройки. Настройки игры сгруппированы по назначению (рисунок 5.2). Первая группа – настройка уровня сложности. В данной игре реализовано 4 уровня сложности: легкий, средний, тяжелый. Их отличие заключается в размере игрового поля и количестве на нем мин. При выборе пользовательского режима активируются дополнительные настройки – пользователь может выбрать одно из трех полей, соответствующих предыдущим уровням, и любое возможное для него количество мин. Вторая группа настроек – звуковые настройки. В них пользователь может включать и отключать фоновую музыку и звуковые эффекты в игре, а также

регулировать их громкость. Третья группа – языковые настройки, в которых пользователь может выбрать язык пользовательского интерфейса игры (русский или английский). Когда настройка игры завершена, пользователь возвращается в главное меню и начинает игру.

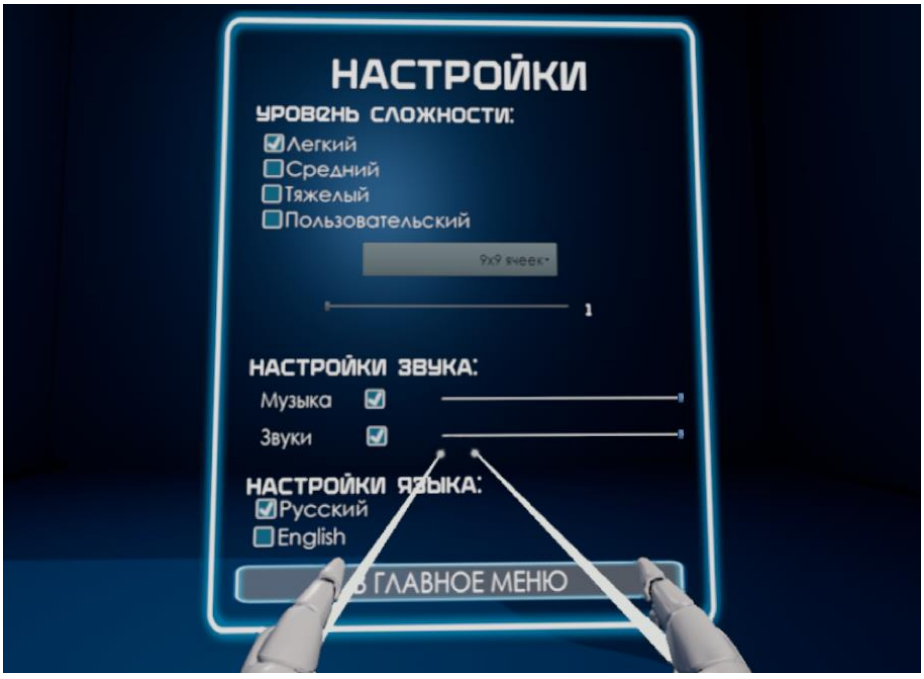

Рисунок 5.2 – Меню настроек игры

Игрок оказывается на игровом поле (рисунок 5.3). В начале игры все ячейки закрыты – отмечены белым полупрозрачным материалом. При открытии ячейки материал становится невидим, над ячейкой загорается цифра. Цифры окрашены в соответствующий цвет для лучшей видимости на поле игры. Все цифры вращаются, плавно перемещаются вверх-вниз и мерцают соответствующими им цветами. Над ячейками, количество соседей-мин для которых равно нулю, цифры не появляются. Помеченные ячейки окрашиваются в красный цвет. Открыть помеченную ячейку можно, открыть одну ячейку дважды нельзя – об этом предупредит звуковой сигнал. В левом верхнем углу отображается таймер игры, в правом верхнем – количество оставшихся мин. Если открывается «заминированная» ячейка, то воспроизводится звуковой и визуальный эффекты взрыва. В этом случае и в случае, если игрок прошел игру, он оказывается на уровне-интерфейсе, где

ему отображается соответствующих виджет с информацией о пройденной игре – ее время и количество открытых мин от общего их количества. Далее предлагается перейти в главное меню.

Рисунок 5.3 – Игровой процесс

В процессе игры пользователь видит в правом нижнем углу карту игры, отображающую небольшой участок поля, на котором находится пользователь. Чтобы посмотреть карту целиком, игрок нажимает соответствующие кнопки на своем контроллере (рисунок 5.4).

Рисунок 5.4 – Карта поля игры

На увеличенной карте игрок видит поле игры в реальном времени – все открытые, закрытые и помеченные ячейки, а также ячейку, на которой он находится в данный момент.

5.2. Анализ качества разработанного ПО

Целью данной работы являлась разработка приложения, соответствующего игровому процессу «Сапёр» в формате ВР. Следовательно, ключевыми критериями качества проделанной работы являются степень соответствия игре «Сапёр» и качество реализации ВР, так как акцент в разработке делался на это. Точную степень соответствия игрового процесса разработанного ПО игровому процессу «Сапёра» установить довольно сложно, так как все функциональные аспекты были сохранены, но их реализация имеет различия в силу того, что игры представлены в разных форматах (двумерное и трехмерное пространство). Но в целом правила остались неизменными, а игра получилась довольно узнаваемой. Что касается качества реализации ВР, то ее можно определить по степени простоты и удобства взаимодействия пользователя с виртуальной средой. В качестве метода оценки данного критерия был проведен эксперимент.

Так как правила и принцип «Сапёра» соответствуют канонической версии игры, то скорость прохождения одного уровня в разработанной в рамках данной работы игре должна стремиться к показателям канонической версии, при одинаковых настройках. Сложное и неудобное управление будет мешать процессу игры, пользователь будет отвлекаться на это, следовательно, скорость прохождения уровня будет расти. В качестве альтернативной версии сапёра была выбрана стандартная игра ОС Windows, другие аналоги, описанные в подразделе 2.2 (Tilesweeper и Crazy Sapper 3D) не подходят для данного эксперимента, так как стратегия игры и структура уровней в них не соответствует каноническим.

Для участия в эксперименте были выбраны пять пользователей различного возраста и с различным опытом игры в «Сапёр». Им предлагалось пройти стандартную игру ОС Windows и игру, разработанную в рамках настоящего проекта, с фиксацией временных результатов. Уровень сложности в обоих случаях был выбран легкий, т.к. некоторые участники эксперимента не имели опыта игры вовсе. Каждый пользователь проходил один уровень по три раза, чтобы наглядно оценить скорость обучения управлению. Результаты эксперимента отображены в таблице 5.1.

Таблица 5.1 – Результаты эксперимента по оценке простоты и удобства использования VR-приложения

№	Возраст	Опыт игры в «Сапёр»	Время прохождения, сек					
			2D-версия			VR-версия		
			1	2	3	1	2	3
1	13 лет	-	160	139	98	203	141	111
2	20 лет	+	115	118	110	175	133	121
3	21 год	+	94	80	105	184	109	104
4	34 года	+	131	131	125	194	139	130
5	45 лет	+	140	132	135	243	199	145

По результатам эксперимента была выявлена зависимость времени прохождения VR-версии от количества прохождений (рисунок 5.5).

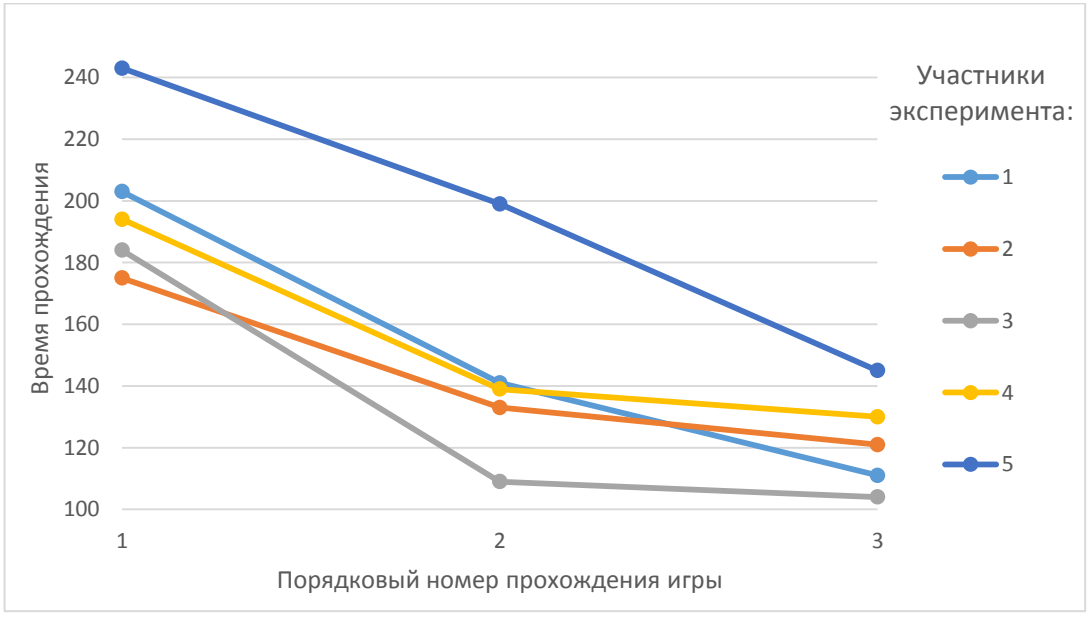

Рисунок 5.5 – Зависимость времени прохождения игры от количества прохождений для VR-версии

Значительное снижение времени прохождения для всех участников эксперимента говорит о том, что во время первых запусков игры происходит привыкание и так называемое «самообучение» новому управлению, а также адаптация к новому формату знакомой игры (для участников эксперимента с опытом игры). Эта адаптация и обучение происходят довольно быстро – все участники значительно сокращают время прохождения с каждым разом. Для участников с опытом игры подобной зависимости для 2D-версии не выявлено.

Для оценки удобства и простоты управления разработанного ПО необходимо сравнить средний результат всех прохождений 2D-версии с результатом 3-го прохождения VR-версии (после прохождения обучения). Результаты сравнения представлены на рисунке 5.6.

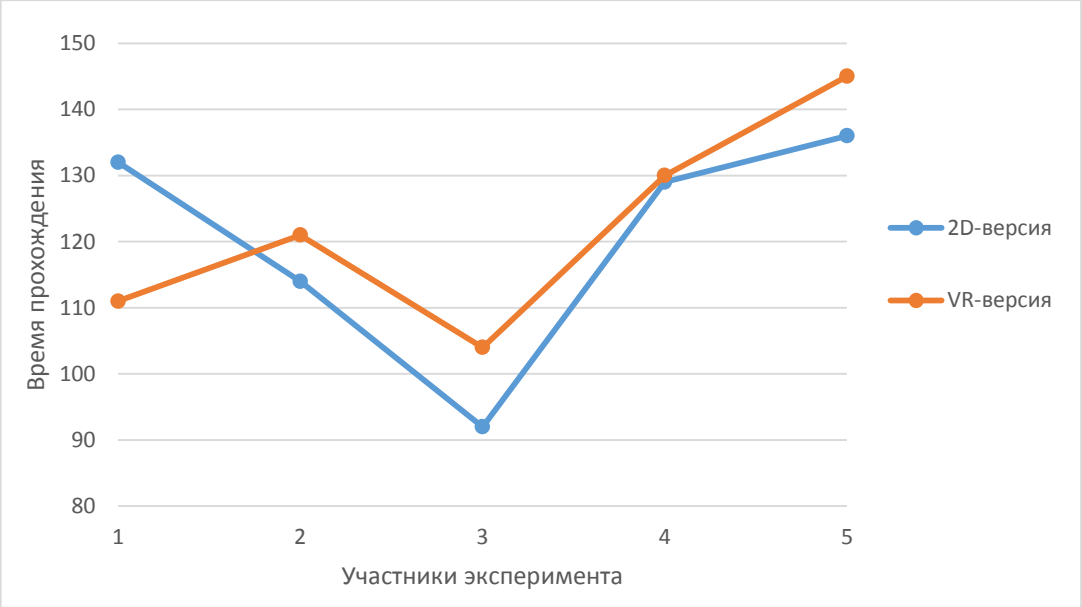

Рисунок 5.6 – Сравнение времени прохождения 2D и VR-версии

По результатам сравнения можно сделать выводы о том, что время прохождения VR-версии в целом близко к времени прохождения 2D-версии, в некоторых случаях превышает его на несколько секунд. Такое превышение может быть связано с тем, что пользователям все же приходится перемещаться через все поле игры, даже посредством телепортации.

Таким образом, результаты проделанного эксперимента говорят об удобстве и простоте управления в игре, а также о быстром периоде обучения, что соответствует казуальному жанру.

Выводы

В данном разделе были описаны и проанализированы полученные результаты работы. Были приведены скриншоты полученного программного продукта с описанием его функциональности. Также было проанализировано качество разработанного ПО. В качестве критерия качества были выделены простота и удобство взаимодействия пользователя с виртуальной средой. В качестве метода оценки данного критерия был проведен эксперимент, в котором сравнивалось время прохождения одного уровня разработанной VR-версии и одного уровня 2D-версии игры «Сапёр» разными пользователями. По результатам сравнения был выявлен факт «самообучения» игроками управлению, а также сделаны выводы о высоком качестве разработанного ПО по выделенному критерию.

6. ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ

Основной целью данной работы является разработка программного продукта, представляющего собой 3D VR-игру «Сапёр». Это версия известной игры «Сапёр», представленная в формате виртуальной реальности: в трехмерном игровом мире с видом от первого лица. В рамка данной работы описывается метод переноса двумерного игрового мира в трехмерное пространство с целью повысить уровень погружения у пользователей. Данная разработка не только сможет привнести вклад в развитие игровой индустрии, а именно ее VR-сегмента, но и поможет развить у ее пользователей пространственное и логическое мышление, быстроту реакции, внимание и другие когнитивные навыки. А также данное приложение позволит любителям «Сапёра» сыграть в него в новом формате и вернет игре интерес у тех пользователей, кто его потерял. Готовое игровое приложение может быть опубликовано на таких платформах-распространителях компьютерных игр как Steam, или любых других, с целью получения прибыли от продажи.

В данном разделе приводится расчёт и обоснование экономической целесообразности данной разработки, а также определяются расходы, затраченные на создание данного программного продукта.

6.1. Расчет расходов на оплату труда и социальные отчисления

Для расчета полных затрат на оплату труда по выполнению данного проекта требуется оценить ресурсы, затраченные на каждый этап разработки. Для этого был составлен детализированный план, соответствующий графику работ. Продолжительность каждого вида работ определяется по фактически затраченному на него времени и измеряется в человеко-днях. Также для

каждого вида работ указывается исполнитель: студент или научный руководитель.

Для каждого исполнителя была рассчитана дневная ставка, которая определяется отношением размера месячного оклада к количеству рабочих дней в месяце, которое было принято за 21. Месячный оклад исполнителей был определен в соответствии с данными за 2020 год, предоставленными порталом sankt-peterburg.trud.com. Средний месячный оклад для профессии инженер-программист составляет 40000 рублей [*], но т.к. работы выполняет начинающий специалист, примем 30000 рублей. Это же значение для профессии доцент кафедры составляет 41617 рублей [*].

Все расчеты были сведены в таблицу 6.1.

Таблица 6.1 – Перечень и трудоёмкость работ с указанием дневной ставки исполнителя

№	Наименование работ	Исполнитель	Трудоемкость, чел/дни	Дневная ставка, руб/день
1	Анализ предметной области и обзор аналогов	Студент	3	1428,57
		Научный руководитель	1	1981,76
2	Выбор средств разработки и платформы	Студент	4	1428,57
		Научный руководитель	2	1981,76
3	Разработка функциональной спецификации	Студент	3	1428,57
		Научный руководитель	2	1981,76
4	Проектирование программного продукта	Студент	4	1428,57
		Научный руководитель	1	1981,76
5	Реализация игрового процесса	Студент	8	1428,57
6	Реализация UI и перемещения в BP	Студент	10	1428,57

7	Создание игрового мира	Студент	6	1428,57
8	Тестирование и отладка	Студент	2	1428,57
9	Подготовка технико-экономического обоснования	Студент	3	1428,57
		Научный руководитель	1	1981,76
10	Оформление пояснительной записки	Студент	13	1428,57
		Научный руководитель	5	1981,76
11	Оформление иллюстративного материала	Студент	3	1428,57
		Научный руководитель	1	1981,76

На основе данных о трудоемкости работ и дневных ставок соответствующих исполнителей были определены расходы на основную и дополнительную заработную плату исполнителей (таблица 6.2). Расходы на основную заработную плату были рассчитаны по формуле 6.1:

$$З_{осн.з/пл} = \sum_{i=1}^{k} T_i \cdot C_i, \quad (6.1)$$

где $З_{осн.з/пл}$ - расходы на основную заработную плату исполнителей (руб.); k –количество исполнителей; T_i - время, затраченное i-м исполнителем выполнение соответствующих ему; C_i - ставка i-го исполнителя (руб./день).

Таблица 6.2 – Расчет основной заработной платы исполнителей

Расходы на дополнительную заработную плату были рассчитаны по формуле 6.2:

$$З_{доп.з/пл} = З_{осн.з/пл} \cdot \frac{Н_{доп}}{100}, \quad (6.2)$$

где $З_{доп.з/пл}$ - расходы на дополнительную заработную плату исполнителей (руб.); $З_{осн.з/пл}$ - расходы на основную заработную плату исполнителей (руб.); $Н_{доп}$ – норматив дополнительной заработной платы (для ВКР принят 14%).

Таблица 6.2 – Расчет заработной платы

Исполнитель	Дневная ставка, руб/день	Общее время работы, день	Основная заработная плата, руб	Дополнительная заработная плата, руб
Студент	1428,57	59	84226,63	11791,73
Научный руководитель	1981,76	13	25762,88	3606,80
ИТОГО			**109989,51**	**15398,53**

Кроме того, необходимо перечислять 30% от расходов на заработную плату исполнителей на социальные нужды, а именно:

- 20% на пенсионное страхование;
- 5,1% на медицинское страхование;
- 2,9% на соцстрахование.

Отчисления на социальные нужды были рассчитаны по формуле 6.3:

$$З_{соц} = (З_{осн.з/пл} + З_{доп.з/пл}) \cdot \frac{Н_{соц}}{100}, \quad (6.3)$$

где $З_{соц}$ – отчисления на социальные нужды с заработной платы (руб.); $З_{осн.з/пл}$ - расходы на основную заработную плату исполнителей (руб.); $З_{доп.з/пл}$ - расходы на дополнительную заработную плату исполнителей (руб.); $Н_{соц}$ – отчисления на страховые взносы (30%).

Результат расчета расходов на социальные отчисления представлен в таблице 6.3.

Таблица 6.3 – Заработная плата и социальные отчисления исполнителей

Исполнитель	Основная заработная плата, руб	Дополнительная заработная плата, руб	Соц. отчисления, руб
Студент	84226,63	11791,73	28805,50
Научный руководитель	25762,88	3606,80	8810,90
ИТОГО	**109989,51**	**15398,53**	**37616,40**

6.2. Расчет материальных расходов

К материальным расходам относят расходы на сырье, основные и вспомогательные материалы, комплектующие изделия, которые необходимы для осуществления проекта.

Общие материальные расходы были рассчитаны по формуле:

$$З_м = \sum_{i=1}^{L} G_i \cdot Ц_i, \quad (6.4)$$

где G – норма расхода на единицу продукции, Ц – цена приобретения единицы материала, L – индекс вида сырья.

В таблице 6.4 представлен расчет затрат на расходные материалы, которые были использованы в рамках реализации данного проекта.

Таблица 6.4 – Затраты на расходные материалы

Наименование	Норма расхода на ед. продукции, шт.	Цена, руб/шт.	Сумма на единицу продукции, руб
Бумага офисная А4 «SvetoCopy»	1	400	400
USB флэш-накопитель Transcend 64 GB	1	580	580
Ручка шариковая Pilot	2	25	50
ИТОГО			1030

Для проведения отладочного и финального тестирования было использовано спецоборудование – стерео-очки для смартфона. Затраты на спецоборудование приведены в таблице 6.5.

Таблица 6.5 – Затраты на спецоборудование

Наименование	Норма расхода на ед. продукции, шт.	Цена, руб/шт.	Сумма на единицу продукции, руб
Очки виртуальной реальности	1	990	990

VR Box 2.0			
ИТОГО			990

Затраты на расходные материалы составили 1030 рублей, на спецоборудование - 990 рублей.

6.3. Расчет амортизационных отчислений

В качестве основных средств при разработке игрового приложения были использованы ноутбук, первоначальная стоимость которого составила 40000 рублей, и смартфон, стоимость которого на момент покупки – 65000 рублей. Следовательно, необходимо учесть и включить в затраты амортизационные отчисления по данным средствам.

Годовая норма амортизации рассчитывается по формуле 6.5:

$$\text{Нα} = \frac{100}{\text{НСС}}\%, \quad (6.5)$$

где HCC – нормативный срок полезного использования средства.

Нормативный срок службы ноутбука составляет 3 года, а смартфона – 2 года, следовательно, годовая норма амортизации равна 33,33% и 50% соответственно.

Амортизационные отчисления за год использования были рассчитаны по формуле 6.6:

$$\text{А}_i = \text{Ц}_{\text{п.н.}i} \cdot \frac{H_{ai}}{100}, \quad (6.6)$$

где $\text{Ц}_{\text{п.н.}i}$ – первоначальная стоимость средства; H_{ai} – годовая норма амортизации.

Тогда можно рассчитать величину амортизационных отчислений по средствам, которые использовались в процессе выполнения разработки, по формуле 6.7:

$$A_{i\text{ВКР}} = A_i \cdot \frac{T_{i\text{ВКР}}}{12}, \quad (6.7)$$

где $T_{i\text{ВКР}}$ – время использования оборудования при работе над ВКР.

По данным из таблицы 6.2 работы с использованием данного оборудования велись в течение 59 дней ~ 2 месяца.

Все расчеты по амортизационным отчислениям сведены в таблицу 6.6.

Таблица 6.6 – Расходы на амортизационные отчисления

Оборудование	Hα, %	Амортизационные отчисления за год (руб.)	Время использования в проекте (мес.)	Амортизационные отчисления (руб.)
Ноутбук	33,33	13332	2	2222,5
Смартфон	50	32500	2	5416,67
ИТОГО				**7639,17**

Амортизационные отчисления составляют 7639,17 рублей.

6.4. Расчет затрат по работам, выполняемым сторонними организациями

Разработка и тестирование игрового приложения проводились с использованием смартфона и стерео-очков при помощи программного продукта Riftcat, которое эмулирует высокотехнологичные VR-устройства, используя гироскоп и другие датчики движения смартфона. Подключение смартфона к ноутбуку проводилось по Wi-Fi сети. Данный способ тестирования использовался на протяжении всей разработки. Wi-Fi соединение также использовалось при создании игрового мира – для использования бесплатного контента, предоставляемого Epic Games, а также при подготовке пояснительной записки.

Расходы на услуги, поставляемые сторонней организацией – Интернет-провайдером «Билайн», представлены в таблице 6.7.

Таблица 6.7 – Затраты по работам, выполняемым сторонней организацией

Наименование	Время использования (мес.)	Стоимость, руб.	Общая стоимость, руб.
Тариф «Стартовый», Билайн	2	400	800
ИТОГО			800

Затраты по работам, выполняемым сторонней организацией составили 800 рублей.

6.5. Расчет накладных расходов

Накладные расходы представляют собой дополнительные к основным расходам. Они могут включать в себя расходы на управление, обслуживание и организацию производства. Накладные расходы рассчитываются индивидуально для каждой организации, на базе которой происходит выполнение работ.

Накладные расходы были рассчитаны по формуле 6.8:

$$Н = \frac{\sum_{i=1}^{2}(З_{доп.з./пл} + З_{осн.з./пл})\cdot Н_{нр}}{100}, \quad (6.8)$$

где $Н_{нр}$ – норматив накладных расходов (примем 20%); i – номер специалиста.

Тогда величина накладных расходов составляет: 25077,61 рублей.

6.6. Расчет совокупных расходов

Все расходы, связанные с данной разработкой были сведены в общую таблицу 6.8.

Таблица 6.8 – Смета затрат на ВКР

№	Наименование статьи	Сумма, (руб.)
1	Расходы на оплату труда	125388,04
2	Отчисления на социальные нужды	37616,40
3	Расходы на материалы	1030
4	Спецоборудование	990
5	Амортизационные отчисления	7639,17
6	Затраты по работам, выполняемым	800

	сторонними организациями	
7	Накладные расходы	25077,61
Итого		198541,22

Себестоимость проекта составила 198541,22 рубль.

Выводы

В рамках данного раздела были подсчитаны величины расходов и составлена смета затрат на разработку – 198541,22 рубля, суммарная трудоемкость составила 72 чел/дня.

Так как для реализации проекта в качестве платформы был выбран SteamVR, то был произведен поиск аналогов разработанного приложения, описанных в подразделе 2.2 данной пояснительной записки, на сервисе Steam. Цена в Steam на оба аналога (Tilesweeper и Crazy Sapper 3D) составляет 4.99$ (367 рублей).

Прогнозирование финансовых результатов данной разработки затруднено в связи с тем, что зависит от достаточно большого количества нестабильных факторов. Однако, может быть смоделирована ситуация публикации приложения на Steam по цене, равной ценам аналогов. В этом случае, после оплаты взноса за публикацию в 100$ (7363 рубля), для окупаемости вложенных средств количество реализованных экземпляров данного приложение должно составить 561 шт. Однако, описанные приложения не являются абсолютными аналогами разработанной игры, так как игровой процесс в них реализован в двумерном пространстве. В Steam, как правило, VR-игры имеют более высокую цену, чем остальные. Так, для реализации настоящего проекта может быть установлена более высокая цена, чем для аналогов, так как оно имеет более сложную техническую реализацию и использует технологии ВР. Следовательно, необходимое для окупаемости количество экземпляров ПО будет тем меньше, чем выше установленная цена.

Срок реализации определенного количества экземпляров будет зависеть от ситуации на игровом рынке и маркетинговой политики.

ЗАКЛЮЧЕНИЕ

В ходе выполнения настоящей работы был проведен поиск существующих программных решений по переносу игровой логики «Сапёра» в трехмерное пространство. По результатам поиска не было найдено ни одного решения, реализующего перенос игрового процесса в пространство виртуальной реальности. Так, целью данной ВКР являлась разработка 3D VR-версии компьютерной игры «Сапёр».

Был проведен поиск и сравнительный анализ найденных средств разработки компьютерных игр. По результатам сравнения был выбран Unreal Engine 4. Для достижения поставленной цели было выполнено проектирование приложения, в ходе которого была разработана функциональная спецификация будущего приложения, его архитектура, сценарии использования UI. Также был разработан алгоритм реализации игрового процесса, основывающийся на структурном разбиении игрового поля на ячейки. В качестве решения по организации перемещения пользователя в пространстве ВР была реализована телепортация. Была проведена отладка и тестирование разработанного приложения с использованием специализированного ПО и устройств, эмулирующих высокотехнологичную VR-гарнитуру.

Таким образом, все поставленные задачи выполнены, цель выпускной квалификационной работы достигнута. Приложение разработано и готово к публикации на онлайн-площадках по распространению игр.

В качестве анализа качества разработанного ПО была проведена оценка удобства взаимодействия пользователя с виртуальной средой, результаты анализа говорят о быстрой самообучаемости управлению и отсутствии сложностей с взаимодействием у пользователей.

Также было проведен расчет экономической целесообразности проекта, по результатам которого сделаны выводы о том, что проект может быть экономически выгоден.

Дальнейшие исследования могут быть посвящены поиску решений по увеличению удобства управления и реализации дополнительных игровых моментов, которые смогут раскрыть весь потенциал возможностей VR-устройств ввода, так как в разработанном на данный момент приложении сделан акцент на нажатие кнопок контроллера, а не на его перемещение в пространстве. За счет этого уровень погружения в игровой процесс может быть увеличен. Также в качестве направления дальнейшего исследования может быть выбрана реализации разработанного приложения в многопользовательском режиме.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Новая философская энциклопедия / В.С. Стёпин, А.А. Гусейнов, Г.Ю.Семигин, А.П. Огурцов. М.: Мысль, 2000. 2659 с.
2. Пат. US 3050870A. Sensorama Simulator / Morton L Heilig. Опубл. 28.08.1962.
3. Пат. US 229067885. A hand gesture interface device / Thomas G.Zimmerman, Jaron Lanier, Chuck Blanchard, Steve Bryson, Young Garvill. Опубл. 1987 г.
4. Пат. US D701206S1. Virtual Reality Headset / Palmer Lackey, Brendan Iribe Trexler, Graham England, Jack McCauley. Опубл. 18.03.2014.
5. SUPERDATARESEARCH.COM / SuperData XR Q1 2020 Update [Электронный ресурс]. Дата обновления: 27.04.2020. URL: https://www.superdataresearch.com/blog/superdata-xr-update (дата обращения: 01.12.2019).
6. Аверин В.А., Маликова Т.В, Кириллов Д.С., Земских Ф.В. Развитие когнитивных навыков с помощью технологий виртуальной реальности // Вестник СПбГУ. Серия 16: Психология. Педагогика. 2017. №2. URL: https://cyberleninka.ru/article/n/razvitie-kognitivnyh-navykov-s-pomoschyu-tehnologiy-virtualnoy-realnosti (дата обращения: 14.05.2020).
7. Virtual Reality Technology / Grigore C. Burdea, Philippe Coiffet. John & Wiley Sons, 2003. 464 с.
8. Slant.co / What are the best game engines for Virtual Reality development? [Электронный ресурс]. Дата обновления: 25.11.2019. URL:

https://www.slant.co/topics/2202/~best-game-engines-for-virtual-reality-development/ (дата обращения: 01.12.2019).

9. Unity [Электронный ресурс]. URL: https://unity3d.com/ (дата обращения: 01.12.2019).
10. What is Unreal Engine 4 [Электронный ресурс]. URL: https://www.unrealengine.com/en-US/what-is-unreal-engine-4/(дата обращения: 01.12.2019).
11. AppGameKit VR [Электронный ресурс]. URL: https://www.appgamekit.com/dlc/vr (дата обращения: 02.12.2019).
12. LibGDX [Электронный ресурс]. URL: https://libgdx.badlogicgames.com/ (дата обращения: 02.12.2019).
13. CryEngine [Электронный ресурс]. URL: https://www.cryengine.com/ (дата обращения: 02.12.2019).
14. Куксон А. Разработка игр на Unreal Engine за 24 часа. М.: Эксмо, 2019. 523 с.
15. Статистика зарплат профессии инженер-программист в Санкт-Петербурге [Электронный ресурс] – URL: https://sankt-peterburg.trud.com/salary/865/3321.html#chart-avgSalaryByYear (дата обращения: 17.05.2020)
16. Статистика зарплат профессии доцент кафедры в Санкт-Петербурге [Электронный ресурс] – URL: https://sankt-peterburg.trud.com/salary/865/76651.html (дата обращения: 17.05.2020)