Министерство науки и высшего образования Российской Федерации
ФГБОУ ВО «Алтайский государственный университет»
Школа развития цифровых компетенций «Digital Up» (цифровая кафедра)

Отчет о выполнении производственной практики по ДПП ПП

«Основы Gamedev и VR-разработки»

# «РАЗРАБОТКА ПОШАГОВОЙ БОЕВОЙ СИСТЕМЫ»

Выполнил студент
направления подготовки
09.03.03 Прикладная
информатика
(профиль «Прикладная
информатика в дизайне»),
группа №8.209-2
Скоснягин Михаил Сергеевич

Руководитель практики:
Старший преподаватель кафедры
культурологии и дизайна
Каратаев Алексей Антонович

Барнаул 2025

**Оглавление**

## Введение

Данная работа производится в рамках производственной практики для курса дополнительного профессионального образования “Основы Gamedev и VR-разработки”.

Цель прохождения практики: получение практических навыков разработки компьютерных игр при решении заданий, представленных компаниями-партнерами программы.

Задачи практики:

1. Проектирование архитектуры игры
2. Реализация базовых игровых объектов и механик
3. Разработка системы ходов и фаз игры
4. Реализация механики действий юнитов
5. Разработка системы выбора карт
6. Реализация определения победителя и UI

Для выполнения производственной практики был выбран один из трёх кейсов-задач от компании-партнёра “Сайберия Нова” (приложение 1).

# ГЛАВА I. АНАЛИТИЧЕСКАЯ ЧАСТЬ

## 1.1 Разбор кейса

Условия, прописанные в кейсе, требуют создания тактической пошаговой системы с тремя фундаментальными механиками. Первая механика - ограничение игры десятью ходами - предполагает разработку особого подхода к проектированию игрового цикла. В отличие от традиционных пошаговых игр, где игра продолжается до определённого условия, здесь необходимо реализовать систему подсчета очков здоровья и механизм определения победителя по итоговому состоянию отрядов. Это создает уникальную динамику, где игроки должны балансировать между агрессивной тактикой и сохранением ресурсов.

Вторая ключевая механика - система действий юнитов - проектируется с учетом специализации каждого типа персонажей. Техническое задание четко разделяет три вида действий, но оставляет пространство для интерпретации их реализации. Особый интерес представляет механика супер ударов, которая вводит элемент прогрессии внутри одной партии. Накопительная система заряда (3 успешные атаки для активации) создает дополнительный стратегический слой, побуждая игроков планировать свои действия на несколько ходов вперед.

Карточная система, как третий столп геймплея, требует тщательного проектирования баланса. В отличие от традиционных коллекционных карточных игр, здесь карты должны иметь временный эффект и ограниченный выбор (2 варианта на ход), что существенно влияет на стратегическое планирование. Особое внимание необходимо уделить дизайну эффектов - они должны быть достаточно мощными, чтобы влиять на ход игры, но не нарушать базовый баланс между юнитами.

## 1.2 Анализ основных игровых механик

Система ходов реализуется как конечный автомат с тремя состояниями: выбор карт, действия первого игрока и действия второго игрока. Каждое состояние требует строгих правил перехода, что обеспечивает

предсказуемость игрового процесса. Счетчик ходов, интегрируемый в GameManager, должен не просто отсчитывать итерации, но и управлять сложной логикой завершения игры, включая обработку ничьих и преждевременных побед.

Механика действий юнитов строится на принципе "выбор одного из трех", что создает тактическую глубину при кажущейся простоте. Каждое действие имеет свои ограничения:

- Перемещение зависит от типа юнита и эффектов карт
- Атака требует соблюдения условий дистанции и линии обзора
- Супер удар доступен только при выполнении условий заряда

Карточная система использует принцип ограниченного выбора для поддержания баланса. В отличие от систем с полной свободой выбора, здесь игроку предлагается только два варианта, что снижает вероятность появления доминирующих стратегий. Эффекты карт требуют тщательного тестирования на предмет синергии и возможных комбинаций, которые могут нарушить баланс.

### 1.3 Подбор и анализ референсов

За основу берутся несколько ключевых референсов. Шахматная механика модернизируется за счет введения асимметричных способностей и системы карт. Из XCOM заимствуется концепция пошаговых тактических боев, но с упрощенной системой укрытий и более четкой специализацией юнитов.

Особое внимание уделяется анализу карточных систем в современных играх. Изучаются механики Hearthstone и Slay the Spire, но с важной оговоркой - карты должны работать как временные модификаторы, а не как постоянные улучшения. Это требует разработки системы баланса, где даже мощные эффекты имеют ограниченное время действия.

### 1.4 Балансировка и математический анализ игровых механик

Балансировка проводится по нескольким ключевым параметрам. Для юнитов рассчитываются базовые показатели эффективности (DPS, выживаемость, мобильность), которые затем корректируются с учетом их специализации. Таким образом, было создано 3 юнита, архетипы которых были заимствованы из классических RPG игр:

- Танк (рыцарь) – мощный, но медленный и имеет короткую атаку, что обусловлено его большим уроном и запасом здоровья. Имеет 20 хп, наносит 6 урона, атаковать может только соседние клетки, перемещается только на соседние клетки. В качестве супер удара у него длинная атака, наносящая удвоенный урон на 2 клетки по прямой.
- Стрелок (лучник) – мобильный юнит с атакой на дальнюю дистанцию. Имеет 15 хп, наносит 5 урона, атаковать может любую клетку по прямой, может перемещаться в любое место в радиусе 2 клеток. Супер удар совершает обычную атаку, но лечит всех юнитов команды на 3 очка здоровья.
- Маг – маломобильный, слабый юнит, но его атака может достать любого врага в любом месте игрового поля. Имеет 10 хп, наносит 4 урона по любой клетке, перемещается также, как и танк. С помощью супер атаки телепортируется в любую выбранную точку карты и наносит 3 единицы урона всем вражеским юнитам.

  Данные характеристики исключают возможность доминирования одного персонажа во всех игровых ситуациях. Также исключена возможность без эффектов карт и суперударов совершать слишком эффективные действия. Например, невозможно быстро победить мага, самого слабого юнита, меньше, чем за 2 хода. Этому препятствует продуманная расстановка и характеристики, например, если вражеский маг и стрелок

Карточные эффекты требуют тщательной проверки баланса, чтобы избежать возможность получить от них слишком весомое преимущество, и наоборот, чтобы избежать наличия «мусорных» карт в игре.

Особое внимание было уделено вероятностным эффектам (критический удар и уклонение). Для них используется система псевдослучайного распределения, которая предотвращает как длинные серии неудач, так и слишком частые срабатывания. Например, шанс крита в 50% фактически работает как гарантированное срабатывание через раз, но с элементами случайности.

Таким образом, были разработаны следующие 8 карт:

- "Ярость" (+1 к урону)
- "Щит" (+2 к защите. При получении урона, 2 его единицы игнорируются)
- "Лечение" (+2 HP всем юнитам команды, но не больше максимума)
- "Скорость" (танк и маг могут совершить одно длинное перемещение, как стрелок)
- "Ультимативный заряд" (+1 к супер удару всем юнитам команды)
- "Уклонение" (50% шанс избежать урона)
- "Критический удар" (50% шанс нанести удвоенный урон)
- "Регенерация" (+5 HP самому раненому юниту команды. Лечит до максимума, если у юнита не хватает меньше 5 хп)

# ГЛАВА II. РАЗРАБОТКА ПРОТОТИПА

## 2.1 Архитектура системы

В основе архитектуры прототипа лежит принцип модульности с четким разделением между компонентами. Центральным управляющим элементом выступает класс GameManager, реализованный по паттерну Singleton, что обеспечивает глобальный доступ к его методам без необходимости поиска объекта на сцене. Данный класс инкапсулирует логику управления игровым циклом, включая обработку переходов между фазами игры (выбор карт, действия игроков), контроль выполнения условий победы и координацию взаимодействия между подсистемами. Важной особенностью реализации является использование перечисления (enum) для определения текущего состояния игры, что позволяет избежать ошибок при смене фаз.

Система игрового поля реализована через класс GridManager, который инициализирует и управляет сеткой 7×7, представленной в виде двумерного массива объектов GridCell. Каждая клетка поля является самостоятельным объектом с прикрепленным компонентом GridCell, наследующим интерфейс IPointerClickHandler для обработки пользовательского ввода. В отличие от простых решений, где клетки выполняют исключительно визуальную функцию, в данном проекте GridCell содержит логику хранения информации о занятости, методы визуальной обратной связи (подсветка) и навигационные свойства. Такой подход позволяет эффективно реализовывать механику перемещения и атаки без дублирования кода.

Класс Hero представляет базовый шаблон для всех типов юнитов и реализует принцип полиморфизма через систему наследования. В нем определены общие для всех персонажей свойства (здоровье, показатели атаки и защиты) и методы (перемещение, атака), при этом специализированное поведение для каждого типа юнита достигается через переопределение виртуальных методов и использование оператора switch в ключевых функциях. Особое внимание уделено системе суперспособностей, где для каждого типа юнита реализован уникальный метод (TankUltimate,

ShooterUltimate, MageUltimate), вызываемый через общий интерфейс UseUltimate.

## 2.2 Ключевые алгоритмы системы

Алгоритм определения доступных клеток для действий представляет собой сложную систему проверок, оптимизированную для работы с сеткой 7×7. В классе ActionSystem реализован метод SetAction, который последовательно анализирует каждую клетку поля, применяя соответствующие фильтры в зависимости от выбранного типа действия. Для перемещения используется проверка CanMoveTo, учитывающая не только расстояние от текущей позиции, но и наличие препятствий. В случае атаки алгоритм дополнительно проверяет принадлежность цели к вражеской команде через свойство isPlayerTeam.

Система генерации карт в классе CardManager использует модифицированный алгоритм Фишера-Йетса для обеспечения равномерного распределения карт между игроками. Особенностью реализации является создание временного пула доступных карт перед каждой генерацией, что исключает повторение одной и той же карты для одного игрока в рамках одного хода. Метод CreateCardsForPlayer включает механизм контроля частоты появления карт, предотвращающий ситуации, когда определенные комбинации эффектов появляются слишком часто.

Обработка суперспособностей реализована через систему накопления заряда, где каждая успешная атака увеличивает счетчик ultCharge. При достижении порогового значения (ultChargeNeeded) активируется соответствующая способность, после чего счетчик сбрасывается. В классе Hero этот механизм интегрирован в метод Attack, что обеспечивает автоматическое обновление состояния без необходимости ручного управления со стороны других систем. Для визуального отображения состояния заряда используется компонент TextMeshProUGUI, динамически обновляемый через метод UpdateVisuals.

### 2.3 Основные связи объектов и элементов UI

Интеграция пользовательского интерфейса с игровой логикой достигается через систему прямых ссылок в инспекторе Unity и глобальный доступ к менеджерам. Класс GameManager содержит ссылки на все ключевые UI-элементы: текстовые поля для отображения здоровья команд (player1HpText, player2HpText), счетчик ходов (turnCounterText) и панели интерфейса (actionPanel, endGamePanel). Обновление этих элементов происходит через метод UpdateUI, который вызывается при любых изменениях игрового состояния.

Система управления действиями реализована в классе ActionSystem и тесно связана с UI через панель кнопок (actionButtonsPanel) (приложение 3). Каждая кнопка (атака, перемещение, суперудар) имеет привязанный обработчик события onClick, который вызывает соответствующий метод SetAction с передачей типа действия. Особенностью реализации является динамическое изменение состояния кнопок - например, кнопка суперудара становится неактивной (interactable = false), когда заряд недостаточен для активации способности.

Отображение карт реализовано через префаб CardObject, который динамически создается для каждой предлагаемой игроку карты (приложение 2). В методе Setup компонента CardObject происходит инициализация текстовых полей (название и описание карты) и привязка обработчика клика, который вызывает применение эффекта через делегат effect. Важным аспектом является немедленное уничтожение объекта карты после выбора, что предотвращает возможность повторного использования.

Визуальная обратная связь обеспечивается через сложную систему подсветки клеток, где каждый цвет соответствует определенному типу действия: зеленый - перемещение, красный - атака, синий - суперудар. Алгоритм подсветки в GridCell учитывает текущий контекст игры, что позволяет избежать визуального наложения состояний. Дополнительные элементы UI, такие как отображение здоровья и заряда ульты над юнитами,

обновляются в реальном времени через методы UpdateVisuals в классе Hero, обеспечивая игроку актуальную информацию о состоянии боя.

## Заключение

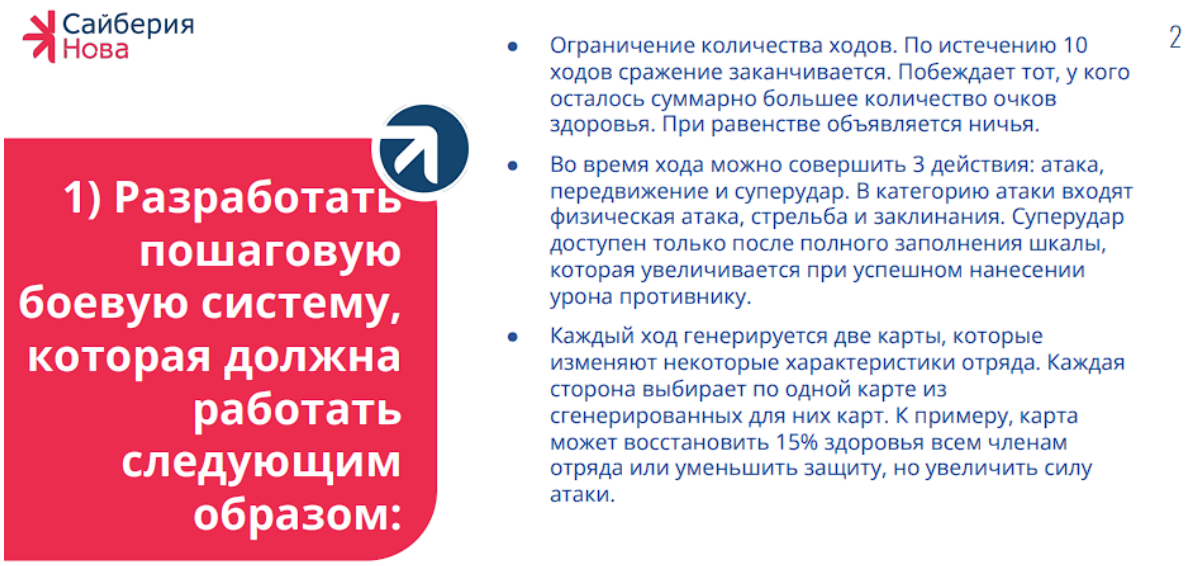
В результате проделанной работы был создан полноценный прототип пошаговой стратегической игры, сочетающий элементы классических тактических игр с современными механиками. Удалось реализовать все ключевые системы: управление юнитами с уникальными характеристиками, пошаговый бой с ограниченным числом ходов, карточные модификаторы и мощные суперспособности. Архитектура игры получилась гибкой – при необходимости можно легко добавить новых персонажей, карты или даже изменить размер поля без переписывания основной логики.

Особую ценность представляет баланс между простотой и глубиной. Правила легко понять, но для победы требуется продуманная стратегия: нужно учитывать позиции юнитов, контролировать заряды суперударов, вовремя использовать карты и предугадывать действия противника. Это делает игру интересной как для новичков, так и для опытных игроков.

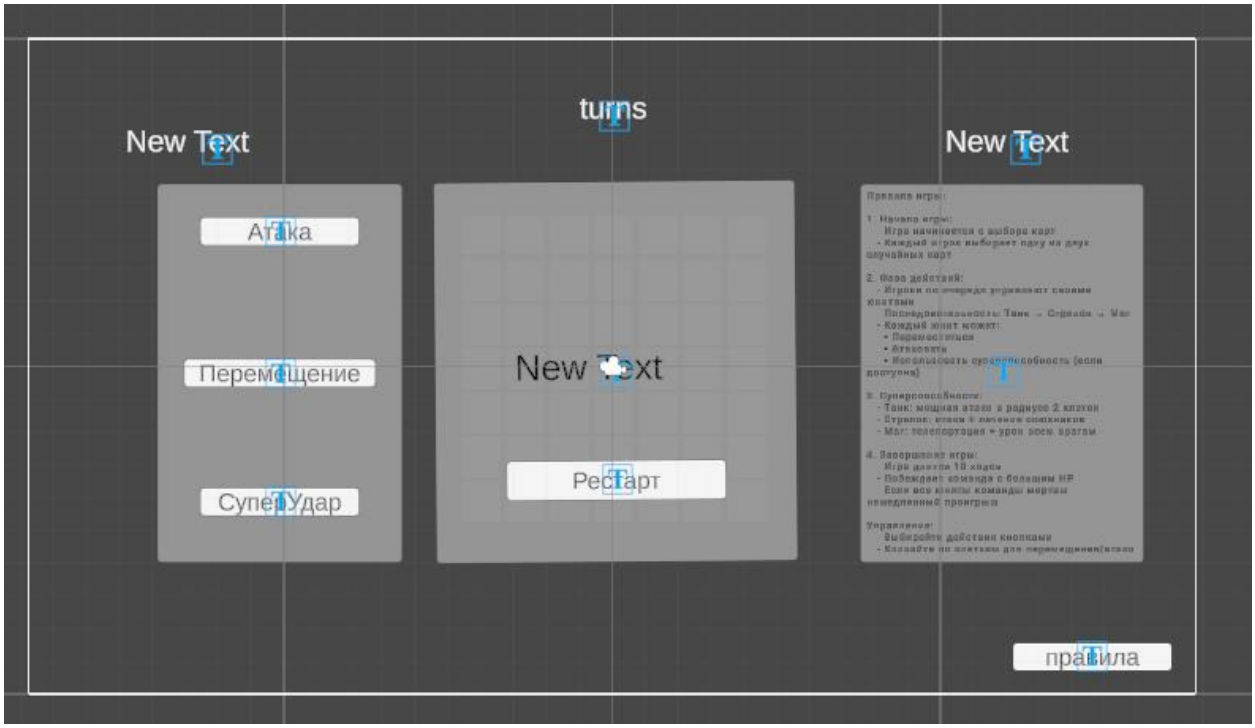
Цели, поставленные в начале выполнения работы были достигнуты. В будущем систему можно расширить – добавить новые типы юнитов, карт и мультиплеер, но уже сейчас она служит отличной основой для дальнейшего развития.

## Приложение

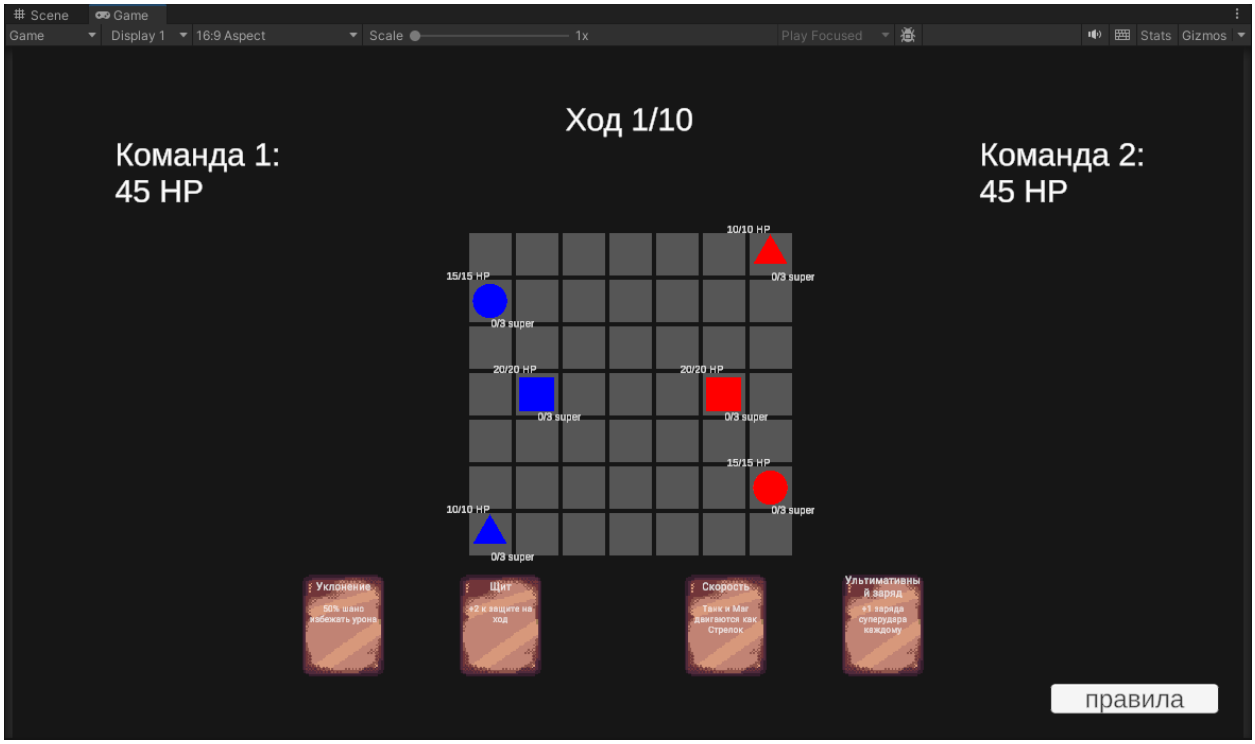
Приложение 1



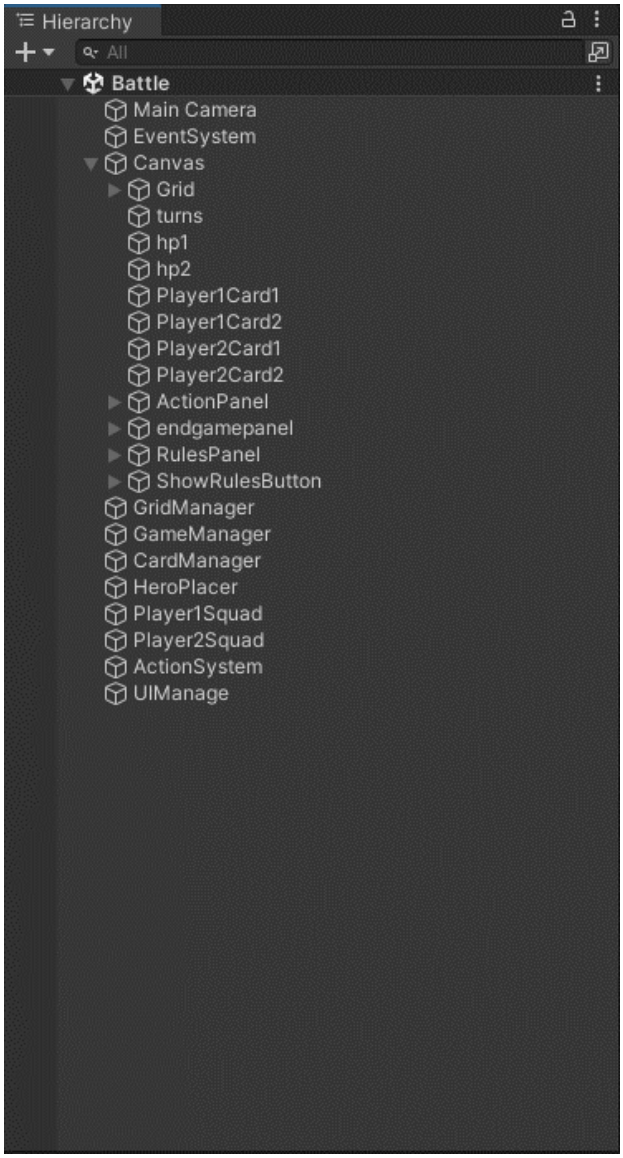

Приложение 2




Приложение 3




Приложение 4



Приложение 5

Видео с демонстрацией геймплея: https://drive.google.com/file/d/19Vx9-nTXrRbg3ICU0xHZjPbL9YoTK7_I/view?usp=sharing

Приложение 6

Репозиторий проекта: https://github.com/Mikhail-Skos/practice