Интеграция сетевых компонентов в шаблон многопользовательской игры на Unreal Engine 4

Студент: Боженко Дмитрий

Научный руководитель: Витюков Ф. А.

Введение

Сетевые компоненты — это элементы сетевой игры.

Сетевые элементы, относящиеся к организации многопользовательской игры:

1. Создание игровых сессий.
2. Авторизация игрока.
3. Создание лобби.
4. Создание таблиц лидеров.
5. Создание системы подбора игроков.

Сетевые элементы, относящиеся к игровому процессу:

1. Возрождение игрока.
2. Реализация изменения очков здоровья.
3. Начисление очков за совершенную ликвидацию.
4. Создание виджета с информацией об игроках.
5. Создание динамического виджета интерфейса.

Постановка задачи

- **Цели работы:** Разработка шаблона многопользовательской игры. Внедрение в него необходимых сетевых элементов. Добавление метода авторизации пользователя через систему Epic Online Services (далее - **EOS**).

- **Задачи:**
  - Изучить принципы работы сетевого программирования в Unreal Engine 4;
  - Добавить в проект следующие сетевые элементы: возрождение игрока, начисление очков, обновление элементов интерфейса (количество очков и здоровья игрока), создание виджета-таблицы со списком игроков;
  - Изучить принцип работы с программным интерфейсом авторизации, предоставляемым EOS;
  - Интегрировать в проект алгоритм авторизации пользователя на уровне приложения.

Сетевое программирование — репликации

Репликация — механизм синхронизации данных об игровом мире между игроками

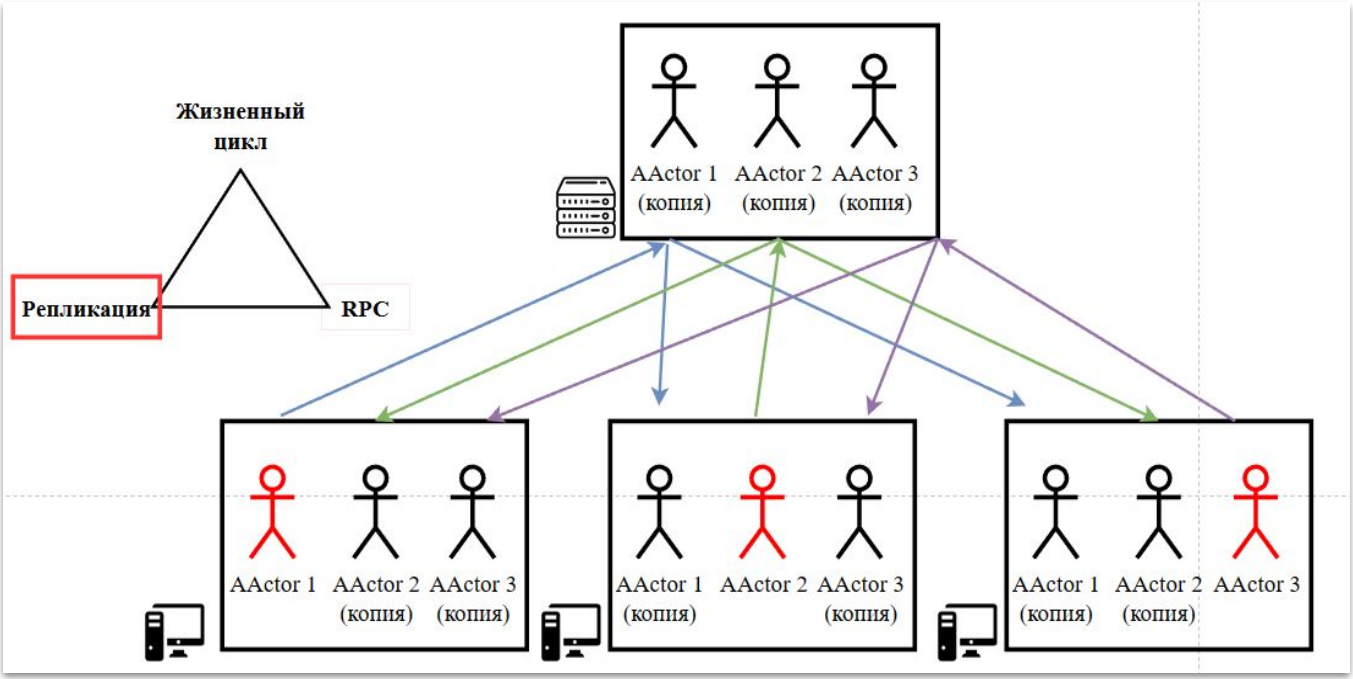


Рис. 1 — Схема репликации в сетевых играх

Сетевое программирование — RPC

Remote Procedure Call (RPC) — функции, которые вызываются с одной машины, а выполняются на другой.

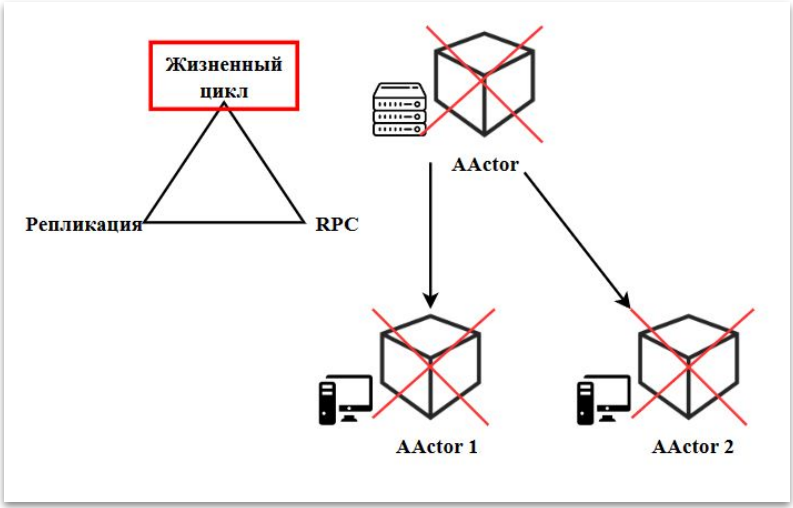


Рис. 2 — Схема жизни объектов в сетевых играх

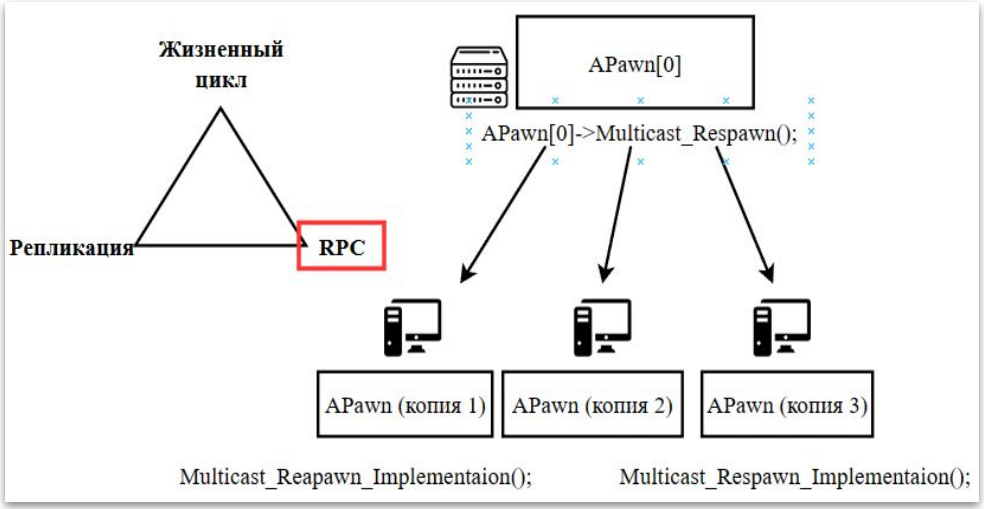


Рис. 3 — Схема вызова RPC

Игровой процесс

Рис. 4 — Игровой процесс по LAN

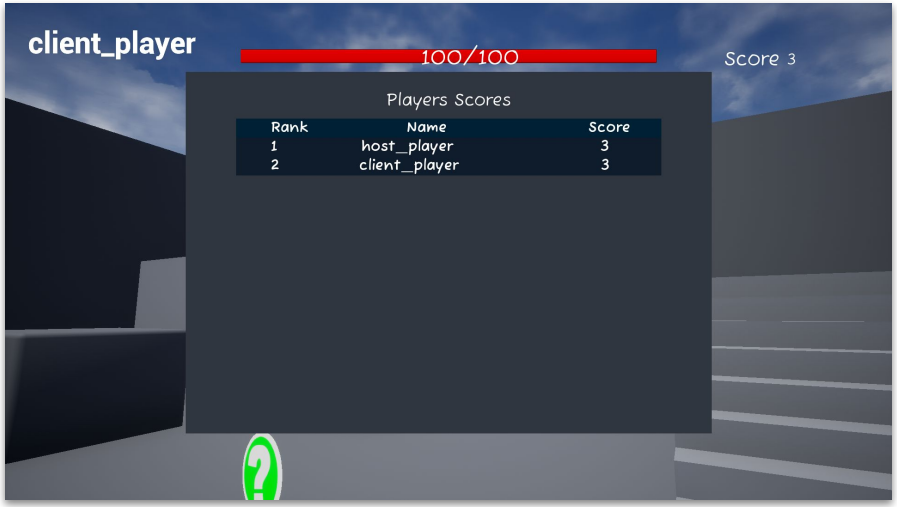
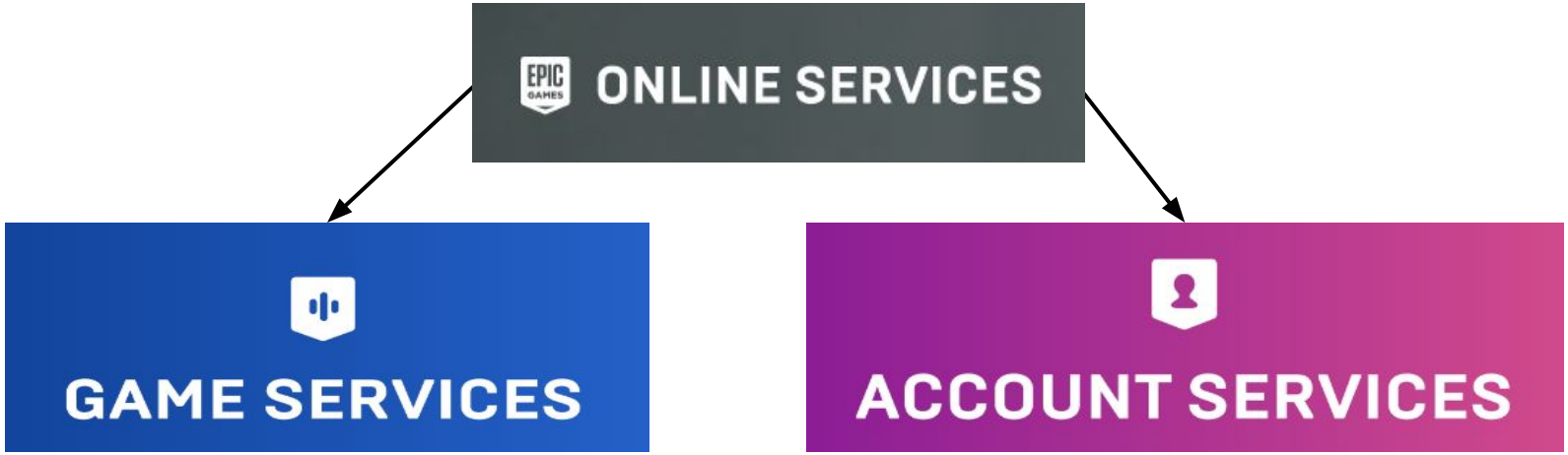


Рис. 5 — Внутриигровой виджет со списком всех игроков

Виджет, отображающий текущую статистику игрока выполнен с помощью средств инструмента *Unreal Motion Graphics (**UMG**)*.

Состояние игрока можно получить с помощью сущности класса *APlayerState*.
Для получения всех сущностей данного класса можно воспользоваться итератором *FConstPlayerContollerIterator*.

Epic Online Services (EOS)



- Lobbies Interface (API создания лобби)
- Leaderboards Interface (API создания таблиц лидеров игры).
- Matchmaking Interface (API создания системы подбора игроков).

- Auth Interface (API для кроссплатформенной авторизация пользователя).
- Friends Interface (API для социальных возможностей взаимодействия с другими игроками).

Авторизация пользователя на уровне приложения (алгоритм)

1. Инициализировать EOSSDK (*EOS_EResult EOS_Initialize(&EOS_Initialize_Options)*)
2. Инициализировать Platform Interface (*HPlatform EOS_Platform_Create(&EOS_Platform_Options)*)
3. Инициализировать Auth Interface (*EOS_HAuth EOS_Platform_GetAuthInterface(HPlatform)*)
4. Реализовать функцию *OnLoginCompleteDelegates (void EOS_CALL CompletionDelegate(const EOS_Auth_LoginCallbackInfo* Data))*
5. Выполнить авторизацию пользователя (вызов функции *void EOS_Auth_Login()*)

Авторизация пользователя на уровне приложения (создание виджетов)

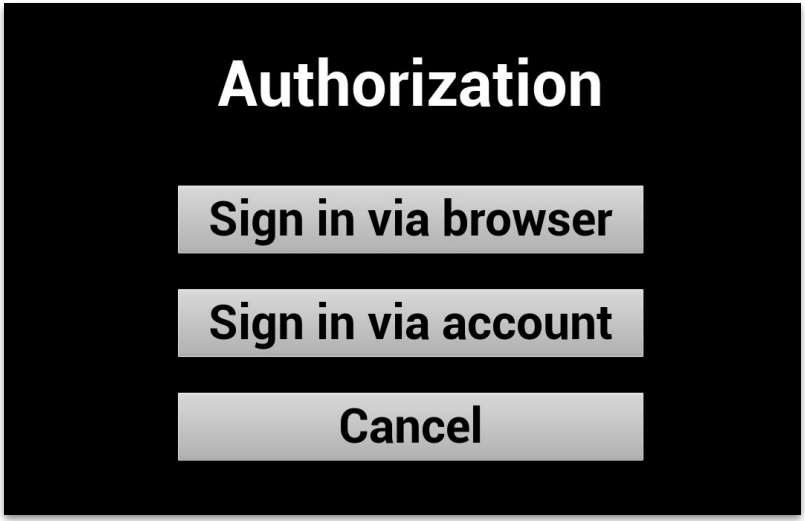


Рис. 6 — Виджет выбора способа авторизации

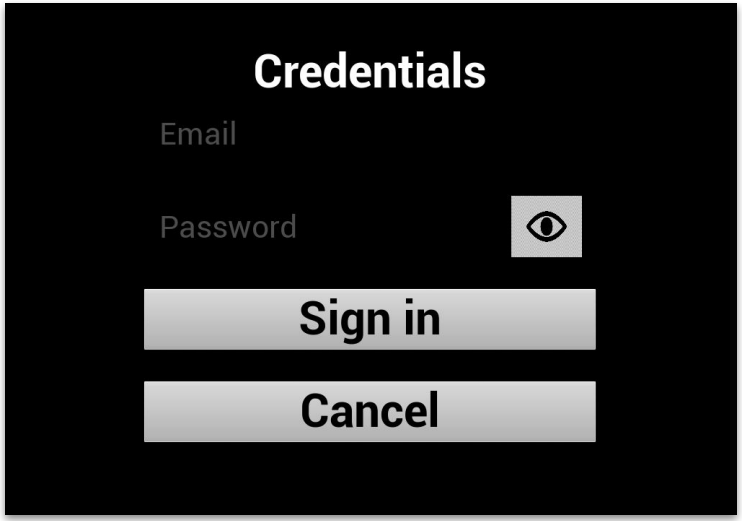


Рис. 7 — Виджет ввода данных от УЗ Epic Games

Разработка всех виджетов выполняется с помощью средств UMG — Unreal Motion Graphics.

Авторизация пользователя на уровне приложения (результат)

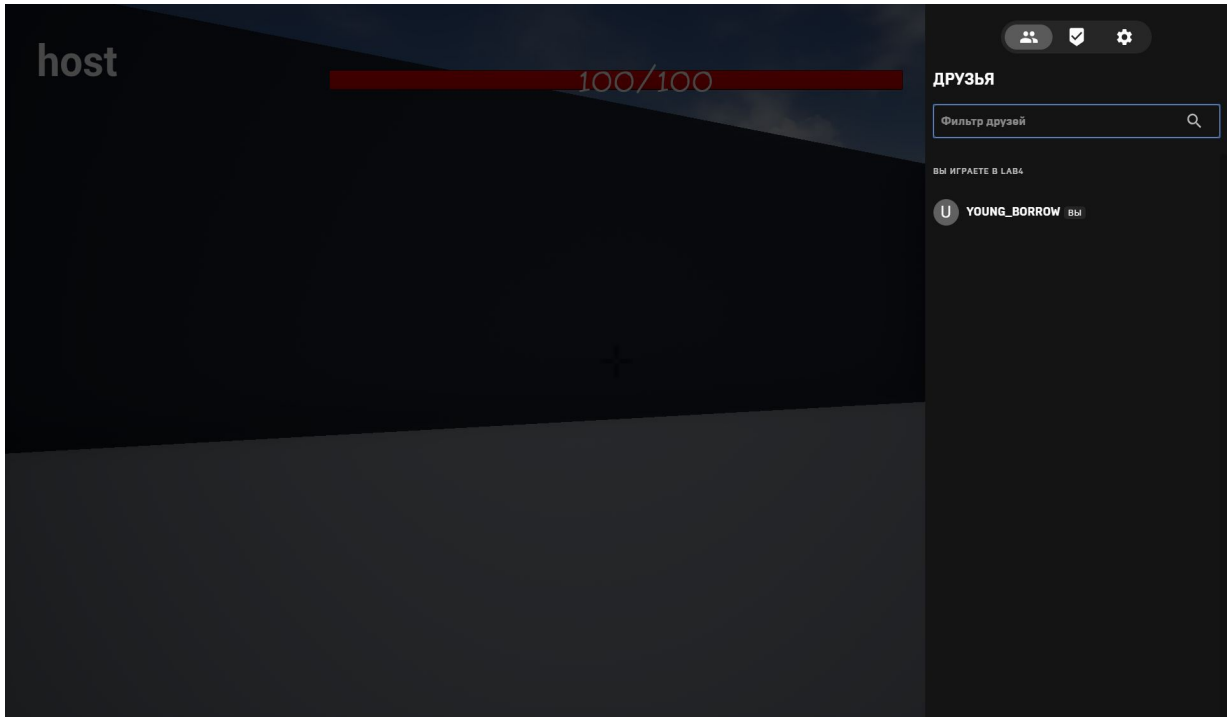


Рис. 8 — Игровой оверлей Epic Games в проекте

Дальнейшая работа над проектом в рамках ВКР

В рамках ВКР необходимо будет решить следующие задачи:

1. Интегрировать в проект программные интерфейсы создания лобби и подключения таблиц лидеров.
2. Изучить тему развертывания выделенных серверов в облаке, реализовать данное развертывание.

Дальнейшая разработка дипломного проекта

В рамках дальнейшей разработки предполагаются меры по улучшению визуальной составляющей:

1. Реализовать правдоподобную анимацию стрельбы, анимацию ликвидации игрока.
2. Оптимизировать проект с точки зрения рендеринга, решить проблему с возможными межсетевыми задержками и проблемами, реализовать соответствующие виджеты.
3. Реализовать web-страницу, которая содержит информацию о продукте, чтобы была возможность опубликовать игру в Epic Games Store.

Выполненные задачи

В результате работы были выполнены следующие задачи:

- Изучены принципы сетевого программирования в Unreal Engine 4.
- Интегрированы сетевые элементы, относящиеся к игровому процессу.
- Освоена работа с плагином Epic Online Subsystem.
- Реализована авторизация пользователя в систему EOS для игры по сети Интернет.
- Изучены и выбраны программные интерфейсы, которые в дальнейшем будут интегрированы в проект в рамках ВКР.