2 СТРУКТУРНОЕ ПРОЕКТИРОВАНИЕ

После анализа всех требований к разрабатываемому проекту, мы переходим к разбиению системы на функциональные блоки. Этот метод позволяет создавать гибкую архитектуру приложения, что в свою очередь позволяет изменять существующие и добавлять новые функциональные блоки, не затрагивая общую работу системы.

Структура проекта состоит из следующих блоков:

- блок игровой логики;
- блок управления;
- блок игрового интерфейса;
- алгоритм чтения аудиофайла;
- алгоритм анализа аудиофайла.

Взаимосвязь основных блоков проекта отображена на структурной схеме ГУИР.400201.304 С1.

2.1. Блок игровой логики

Данный блок содержит в себе базовую логику игровых правил и взаимодействий игрока с игровыми объектами. В блоке игровой логики описываются основные логические элементы игры, такие как условия проигрыша, подсчёт очков игрока и возможные взаимодействия игрока с игрой и игры с игровым персонажем. В данный блок так же входит установление взаимосвязей между другими блоками данного проекта. Для реализации данного блока будет использован язык визуального программирования Blueprints.

2.2. Блок управления

Для осуществления управления игровой пешкой необходимо создать блок, отвечающий за считывание нажимаемых клавиш, и преобразующий эти нажатия в вызовы функций и событий из блока игровой логики. Нажатия должны считываться как с клавиатуры, так и с других возможных устройств контроля. В данном блоке должна быть реализована реакция на нажатие определённых, заранее установленных разработчиком клавиш, отвечающих за исполнение игровой пешкой конкретных действий. Для реализации данного блока будет использован язык визуального программирования Blueprints.

2.3. Блок игрового интерфейса

Блок игрового интерфейса отвечает за интерфейс главного меню игры, интерфейса для взаимодействия с аудио файлами, а так же за интерфейс самого игрового процесса, счётчика очков и других элементов индикации и

вывода информации пользователю. Для реализации данного блока будет использован язык визуального программирования Blueprints.

2.4. Алгоритм чтения аудиофайла

Блок алгоритма осуществляющего чтение заголовка и блока данных аудиофайла. Первоначальной задачей алгоритма является определение формата файла, валидация событий, приводящих к возникновению ошибок при чтении и открытие файла в режиме чтения. Аудио файл должен соответствовать формату .wav, так как данный формат файла используется для хранения несжатого аудио сигнала в импульсно-кодовой модуляции. После открытия файла, главными задачами блока будут являться, чтение заголовка, инициализация структуры для сохранения данных из заголовка аудио файла, таких как количество каналов, частоту дискретизации, аудио формат, наличие и тип кодировки, количество байт для хранения одного сэмпла, а так же общий размер файла без учета первых 16 байт. Данная информация необходима для корректного чтения блока данных конкретного аудио файла. Так же полученная информация будет использована для анализа композиции в блоке анализа аудиофайла. Данный алгоритм будет реализован с использованием языка программирования C++.

2.5. Алгоритм анализа аудиофайла

Блок алгоритма анализа аудиофайла. Данный алгоритм отвечает за обработку и анализ блока данных. Используя информацию, полученную в заголовке файла в алгоритме чтения файла, производится быстрое преобразование Фурье с использованием окна Гаусса. Использование окна Гаусса позволит избавиться от возможного появления шумов, после применения БПФ функции. Входными значениями алгоритма являются массив с амплитудно-временными значениями, а так же размер данного массива. Выходным значением является массив с амплитудно-частотными значениями, отображающими перепады амплитуд конкретных диапазонов частот в определённый промежуток времени. На основе данных значений производится анализ и вычисление ритма музыкальной композиции, который в дальнейшем отправляется в блок игровой логики. Данный алгоритм будет реализован с использованием языка программирования C++.