Design Specifications

Firmware Infirmary

Version 1.1

Defined By:

Billy Carroll, Jovan Kouakou, Mike Mariano, Nick Pelletier, Ryan Marcovecchio, Tom Card

Advisor:

Jeff Salvage

Stakeholder:

Drexel University

Table of Contents

Table of Contents **2**
3.1 Architecture Diagram **4**
3.2 Technology Used **5**
3.3 Component Diagram **6**
3.4 Sequence Diagrams **6**

2. Introduction

2.1 Purpose

The purpose of this design document is to display and explain the architecture design of Mudsofts game. The architectural diagrams outlined here align with the requirements stated in our SRS(Software Requirements Specification).

2.2 Scope

The scope of this document outlines the architecture that was used in the development of our game including outsourced packages, software, and resources used the development of our game as well as the structure of our code base.

2.3 Design Goals

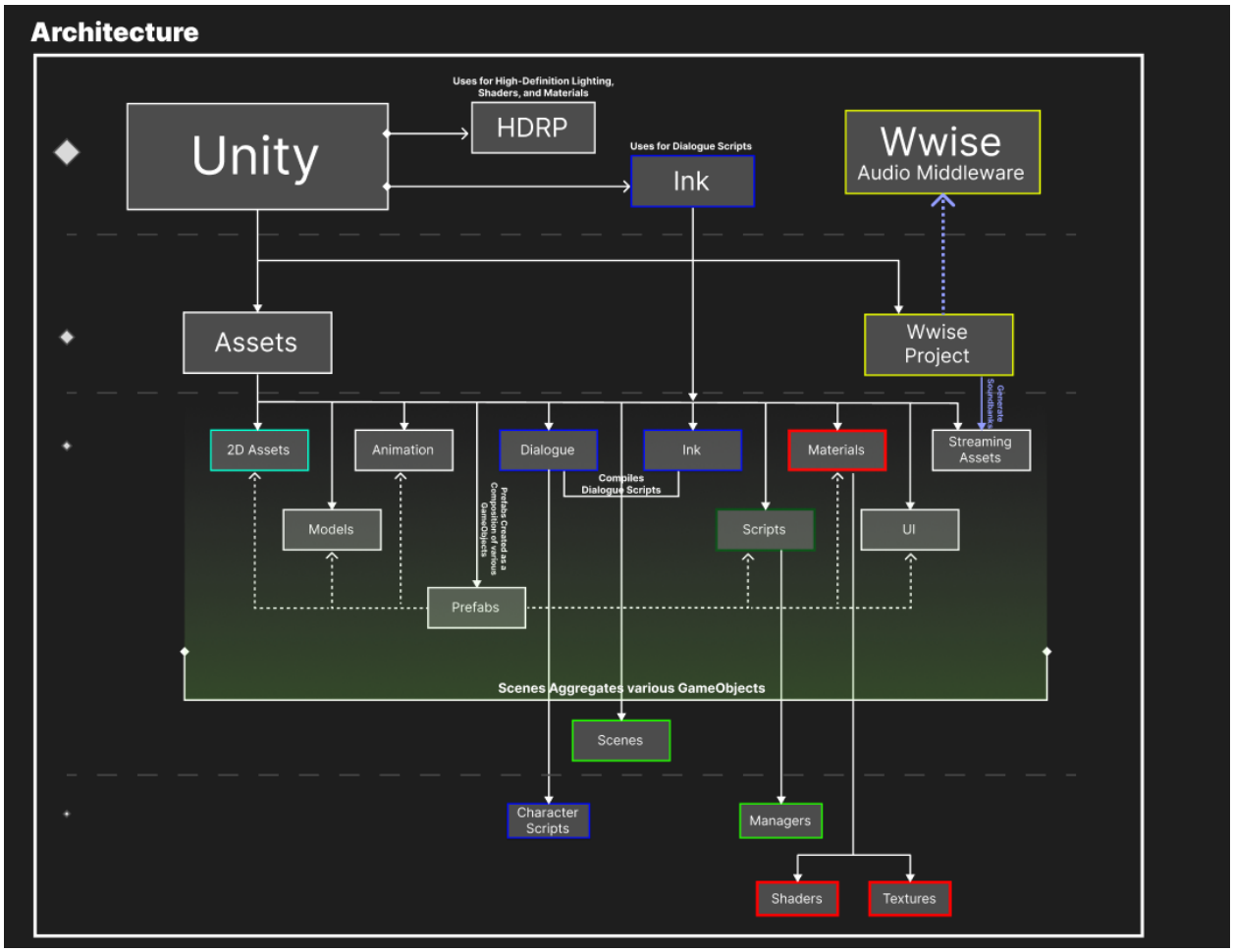
Streamline the creative flow and expected workflow of our game's mechanics. Due to the importance of our backend system and states, the goals of these diagrams were to better understand and create a strong design, programmatically, for our game.

2.4 Definitions

GameObject: A unity object that is referenced within code and is a general denoter for objects that have been placed into the scene.
Prefab: A unity special component that allows fully configured GameObjects to be saved in the project for reuse.

3. Architecture Overview

3.1 Architecture Diagram

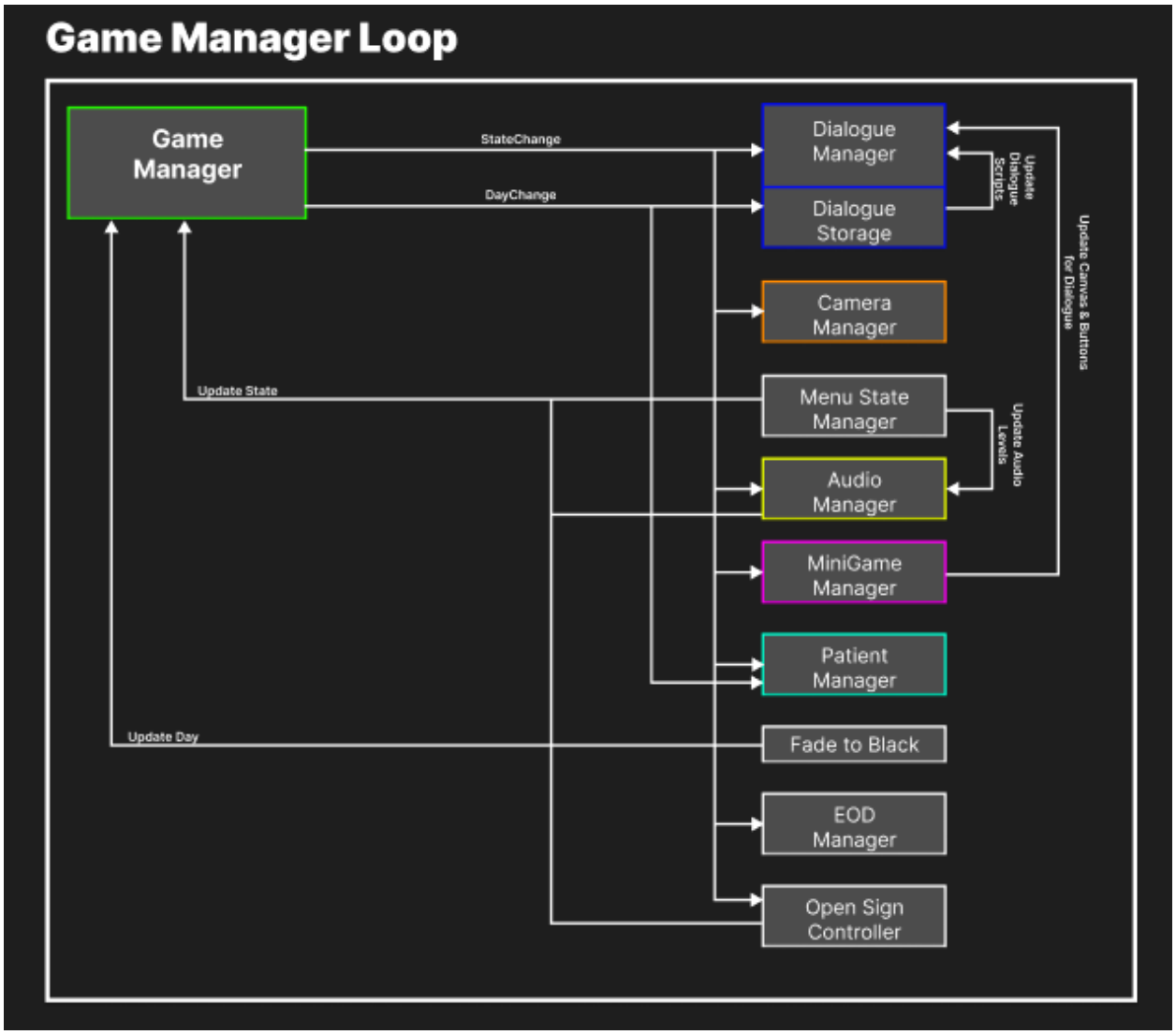


The overall design of our code base for our game can be seen above broken down by both color and level hierarchy. The game was developed through Unity which broke down into our different assets which can be broken down further into individual scripts, animations, models, and prefabs.

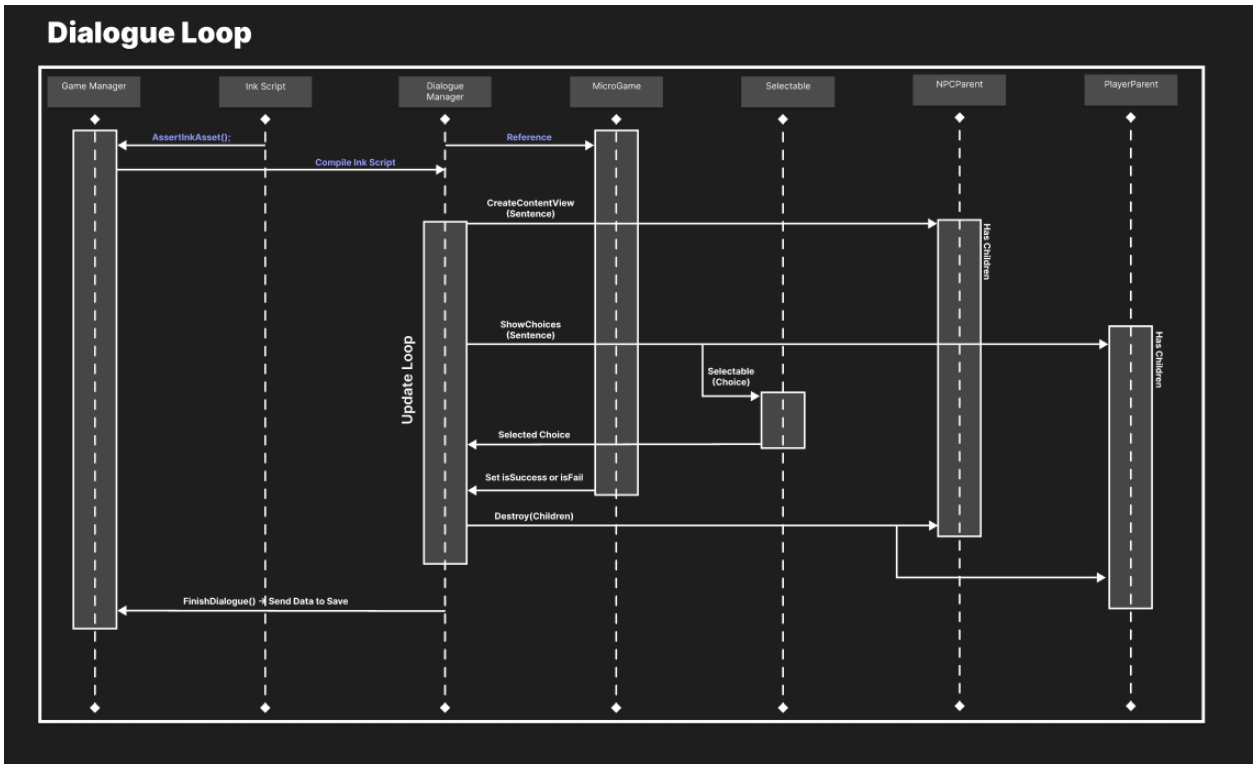
3.2 Technology Used

The development of our game utilized Unity as our main development platform while using Perforce as our storage and version control for development, the game itself utilized C# for the creation of functional scripts and managers. The use of Wwise with Unity was for the purpose of audio implementation into our game.

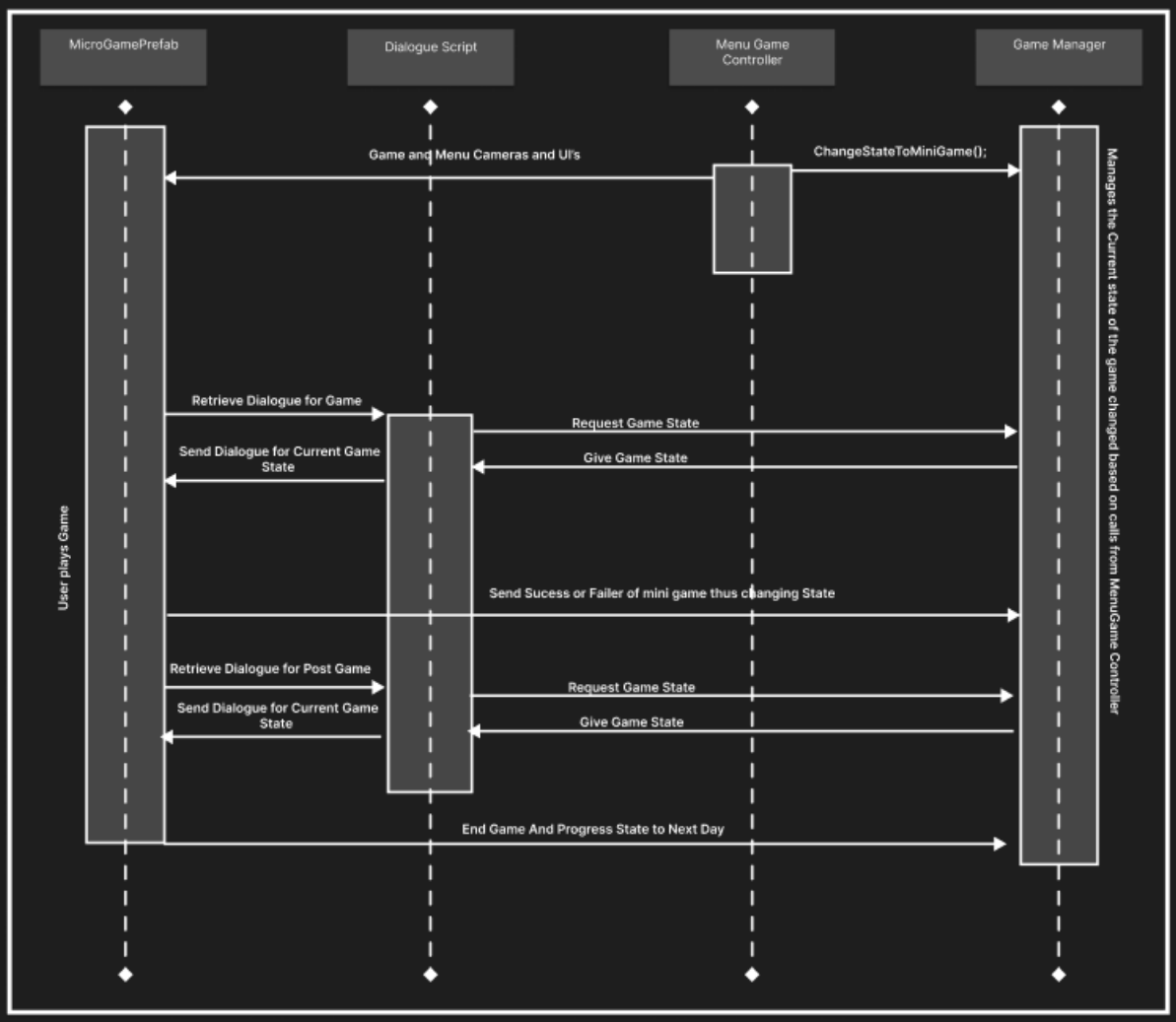
3.3 Component Diagram

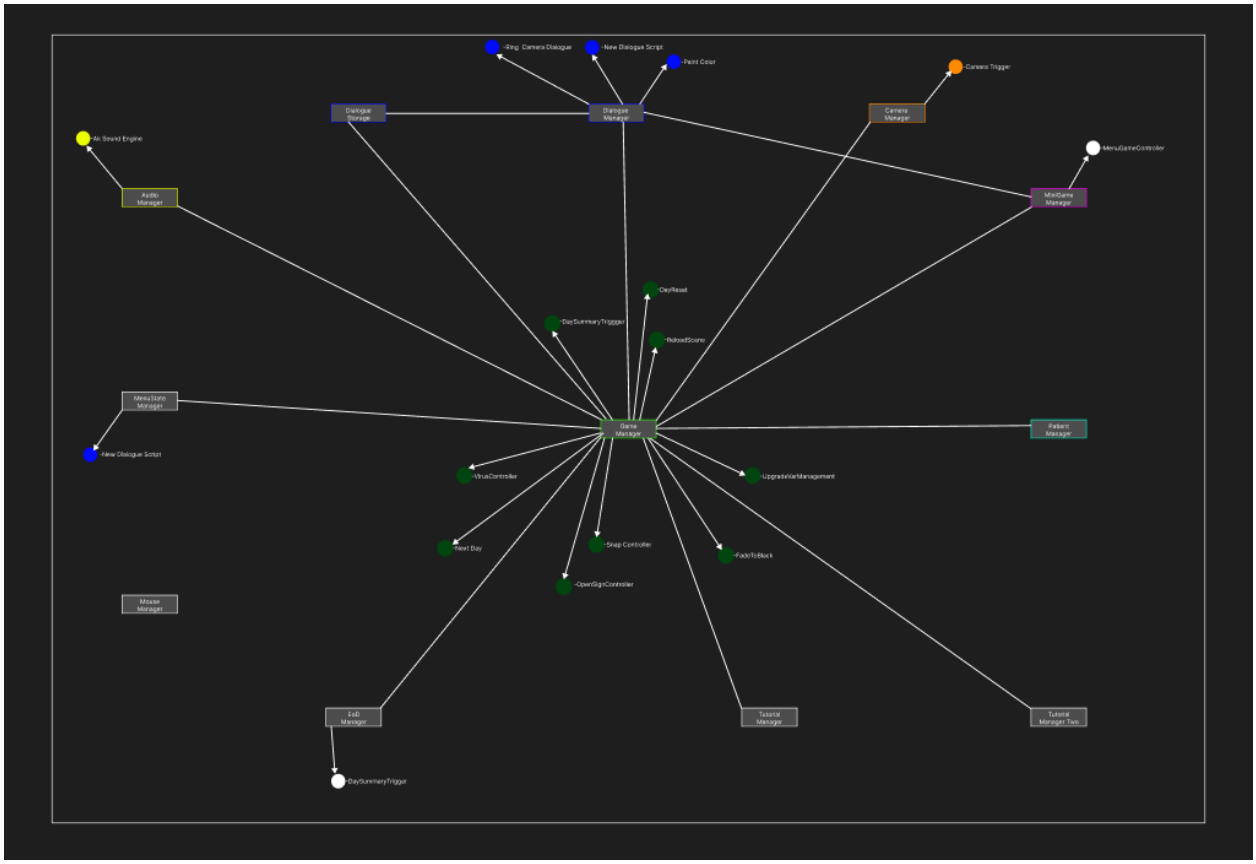


3.4 Sequence Diagrams

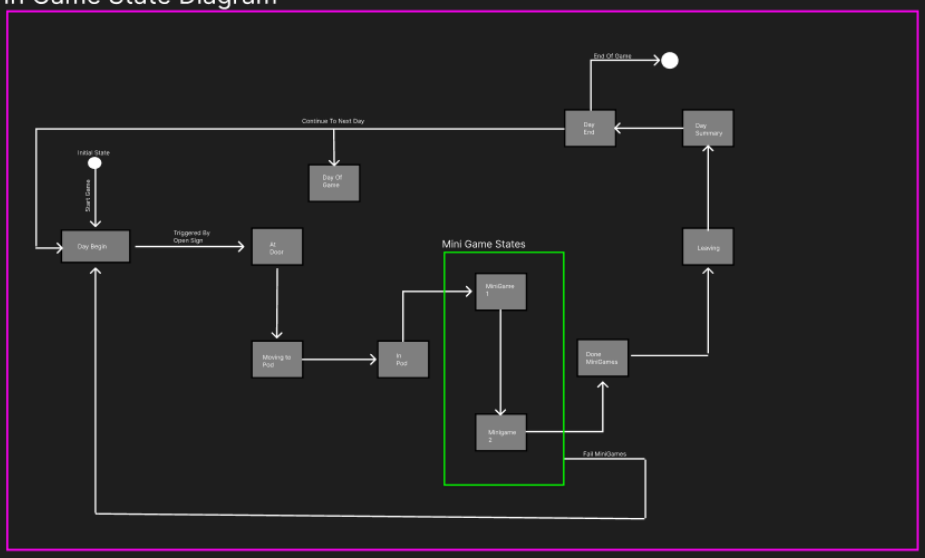
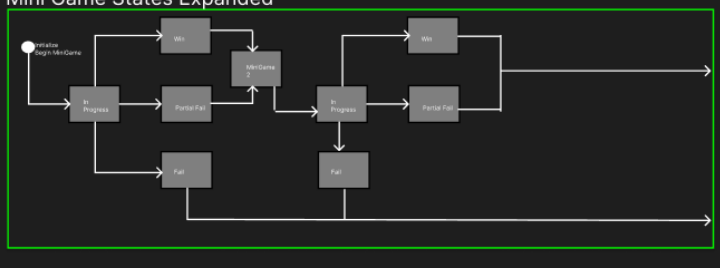

Dialogue Loop
Game Manager
Ink Script
Dialogue Manager
MicroGame
Selectable
NPCParent
PlayerParent
AssertInkAsset();
Compile Ink Script
Reference
CreateContentView (Sentence)
Has Children
ShowChoices (Sentence)
Update Loop
Selectable (Choice)
Selected Choice
Set isSuccess or isFail
Destroy(Children)
Has Children
FinishDialogue() -> Send Data to Save

MicroGame Loop




-Sting- Camera Dialogue
-New Dialogue Script
-Point Color
-Camera Trigger
Dialogue Storage
Dialogue Manager
Camera Manager
-As Sound Engine
-MenuGameController
Audio Manager
MiniGame Manager
-DayReset
-DaySummaryTriggger
-ReloadScene
Menu Store Manager
Game Manager
Patient Manager
-New Dialogue Script
-VirusController
-UpgradeVarManagement
-Next Day
-Snap Controller
-FadeToBlack
-OpenSignController
Mouse Manager
EoD Manager
Tutorial Manager
Tutorial Manager Two
-DaySummaryTrigger

In Game State Diagram



Mini Game States Expanded