ALPHA\_RCRacing / Input & Output

Architecture/Design Document

**Table of Contents**

1 Introduction.. 3

2 Design Goals.. 3

3 System Behavior.. 3

4 Logical View... 3

4.1 High-Level Design (Architecture of the Entire system)… page 4

4.2 Mid-Level Design of the Input & Output Module... page 5

4.3 Detailed Class Design of the Input & Output Module… page 6

5 Process View of the Input & Output Module... page 7

6 Use Case View... page 8

Change History

**Version:** 0.1

**Modifier:** Alexander Achorn

**Date:** 03 / 18 / 2022

**Description of Change:** Module Design Document started.

**Version:** 0.2

**Modifier:**

**Date:**

**Description of Change:**

1. **Introduction**

This document describes the architecture and design for ALPHA\_RCRacing, a game being developed by Inertial Sketch. ALPHA\_RCRacing is a Multiplayer RC Car Game where player compete against each other in a competitive racing track filled with jumps, turns and Power Ups.

The purpose of this document is to describe the architecture and design of the Input & Output Module application in a way that addresses the interests and concerns of all major stakeholders. For this application the major stakeholders are:

* Developers;
* Project Manager.

1. **Design Goals**

The design priorities for the Input & Output system are:

* The design should allow for intuitive user input
* The design should allow for different outputs (VR vs Pancake/ Flat)
* The design should avoid making the player uncomfortable (i.e. nausea)

1. **System Behavior**

The Input & Output module is built primarily of settings that exist in the RacingPawn but will likely also exist in a DummyPawn used for the Main Menu. These settings must be adaptable to different output methods whether it be traditional gameplay or through the use of VR. The input must also facilitate the navigation of menus and handing control to and from the game and the menus.

1. **Logical View**

The logical view describes the main shared components, attributes and switches of the system. This includes modules, the static relationships between modules, and their dynamic patterns of interaction.

In this section the modules of the system are first expressed from a macro perspective and progressively goes to a micro perspective to view the detailed sequences and components.

* 1. **High-Level Design (Architecture of the Entire system)**
* RacingPawn System is the main system and handles the car movements, current power up and various effects.
* PowerUp System knows its mesh and collision sphere as well as its cooldown and despawn timer. Its children have unique effects.
* UI System is used to indicate crucial information like the player’s current power up.
* Menu System handles the selection of car, map and handles the race creation.
* InputComponents system handles the controls for PC, Console and VR.
  1. **Mid-Level Design of the Input & Output Module**

Diagram

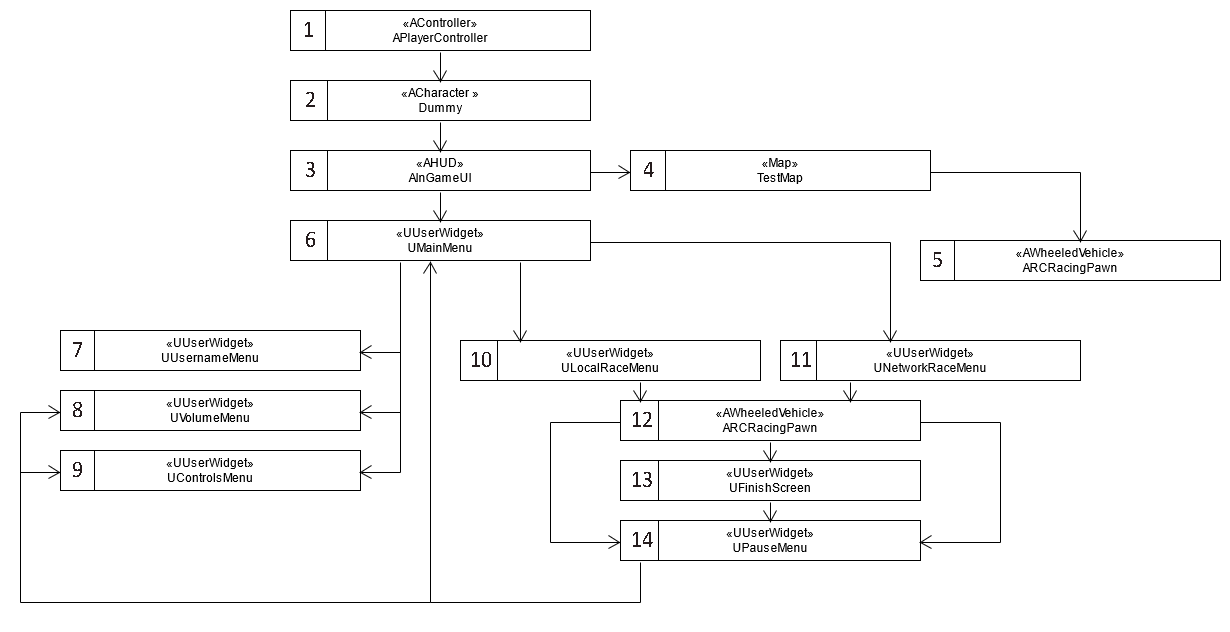
Description automatically generated

* 1. **Detailed Class Design of the Input & Output Module**

Text, letter

Description automatically generated

1. **of the Input & Output Module**



The Input, Output sequence begins at the start of the game when the MenuMap checks if the Player is using VR or not. If using VR the TestMap will be loaded and the RCRacingPawn will set a variety of Camera settings specifically for VR.   
  
If not using VR the PlayerController connected to the Dummy Player will call InGameUI’s ShowMainMenu. From that point the controls are handed over to the UI as the MainMenu is added to the viewport. The player then can navigate through the menus to each submenu. If they proceed to Network or Local play and select a map said map will load.   
  
After the new map is loaded the Dummy player is now swapped for a RCRacingPawn who will handle controls and menu interaction. If a race is finished or the player used the pause menu the sequence can return to the main menu to loop back and pick another map or mode.  
  
While the sequence is nearly identical to the Menu sequence and contained within other Classes the Input and Output module and loop are integral to the smooth integration of VR as adaptations will need to be made throughout rather then in a single class.

1. **Use Case View**

For this release, all Menus & UI only work with a mouse. In future releases, keyboard and gamepad support should be implemented as part of this module. VR controller support should in theory work for driving but testing was limited by local hardware. Due to menu limitations VR mode automatically loads the test map as you can’t open the menus at all while using VR.

To help facilitate easy navigation a controls menu can be accessed from the Main and Pause menus. This lists all the common controls. Vive, Index and Oculus controllers should generally be mapped the same as the gamepad with the exception of Vive wands using their trackpad in rather then the joystick on other platforms.

The UI system is configured to automatically take control of the controls when focused and to enable a cursor for ease of navigation.