Virtual Reality---Texting While Driving

Jacob Wheeler: Major in SE
Nathan Christiansen: Major in SE
Nicholas Kapty: Major in SE

Course Instructor: Xiaocong Fan
Faculty Advisor: George Dudas

Industry Sponsor: Erie Insurance
Project Mentor: Matthew Panetta
IT Apprentice
Project Proposer: MaryJo Ingalls
Supervisor Enterprise Content Management

A capstone project report submitted to the faculty of
The Computer Science and Software Engineering Department
Penn State Erie, The Behrend College

February 2017
(Version 2.5)

CSSE Technical Report Series: CSSE-BD-Class2017-001

Contents

1. Abstract

Erie Insurance currently works with its agents to help them display the dangers of distracted driving to their policyholders. This can often be very difficult for agents to do since the user is not able to experience the consequences of distracted driving for themselves in a safe way. In order to help solve this problem for the agents, we are creating a virtual reality experience to demonstrate how distracted driving can affect the policyholder. This virtual reality experience will utilize the Unity 3D engine and the Google Cardboard SDK to give the policyholder different scenarios in which they will have to make decisions influencing their outcome. This virtual reality experience will help the policyholder to understand how they can influence dangerous driving activities as well as to help stop them.

2. Report Revision History

2.1 Changes in Version 1.5

In this version, we have made the changes recommended to us by our advisor. We have added a new user requirement and functional requirement detailing more information regarding the specific tasks that the AI driver should perform. The use case mapping diagram has updated as well. Along with that, we have changed the name of our use case “Begin Experience” to “Experience Loop” to make more sense. References are now available and are used in section 5 to further explain our exploratory studies.

2.2 Changes in Version 2.0

In this version, we have added our initial designs for the architecture, structure, interface, and behavior of the system. We have changed our architecture to the component-based architecture, which more accurately captures the way Unity objects build off each other to create the overall system. We have added and updated our requirements based off feedback from advisors and industry mentor. We have created test cases for our system, as well as the execution history. We have added the steps to set up the development environment and testing environment, build for the target platform, and install to the end user’s device.

2.3 Changes in Version 2.5

In this version, we have made further changes recommended to us by our faculty advisor and industry mentor. We have modified the layout of the report in section 6.2 to better organize the descriptions that go with each individual image. We have also updated section 6.3 to be contained within one page for further formatting improvements.

3. Problem Statement

3.1 Business Background

Erie Insurance is a Fortune 500 insurance company employing thousands of people. Erie Insurance has been a figure in the insurance world for 90 years, and currently serves over 4 million customers in 13 states. They utilize and manage smaller agencies to deal directly with customers, selling them auto, home, life, and business insurance.

With the rise of technology, distracted driving has become more of a risk than ever before. As Erie Insurance invests in protecting people, they are taking the initiative in informing families about the dangers of driving while distracted.

3.2 Needs

Currently, it is very difficult to display the dangers of distracted driving to a younger generation in a way that engages them. Erie Insurance is seeking an innovative solution in order to solve this problem.

3.3 Objectives

This project aims to utilize virtual reality technology to create an immersive experience that engages users of all ages. The application will be distributed to agents around Erie's footprint and will effectively capture the younger audience.

4. Requirements

4.1 User Requirements

4.1.1 Glossary of Relevant Domain Terminology

Virtual Reality (VR) – A simulation of a three-dimensional environment

Cardboard – Google's SDK created for smartphone devices

Headset – A head mounted device that displays virtual reality devices

Scene – A Unity scene is an aggregation of components that can be executed on its own

4.1.2 User Groups

User – Any person engaging in our experience

4.1.3 Functional Requirements

4.1.3.1 Project Scope (Use Case Diagram)

Figure 4.1 displays the system's use case diagram. This gives a layout of the main user interactions that can occur as they use the system.

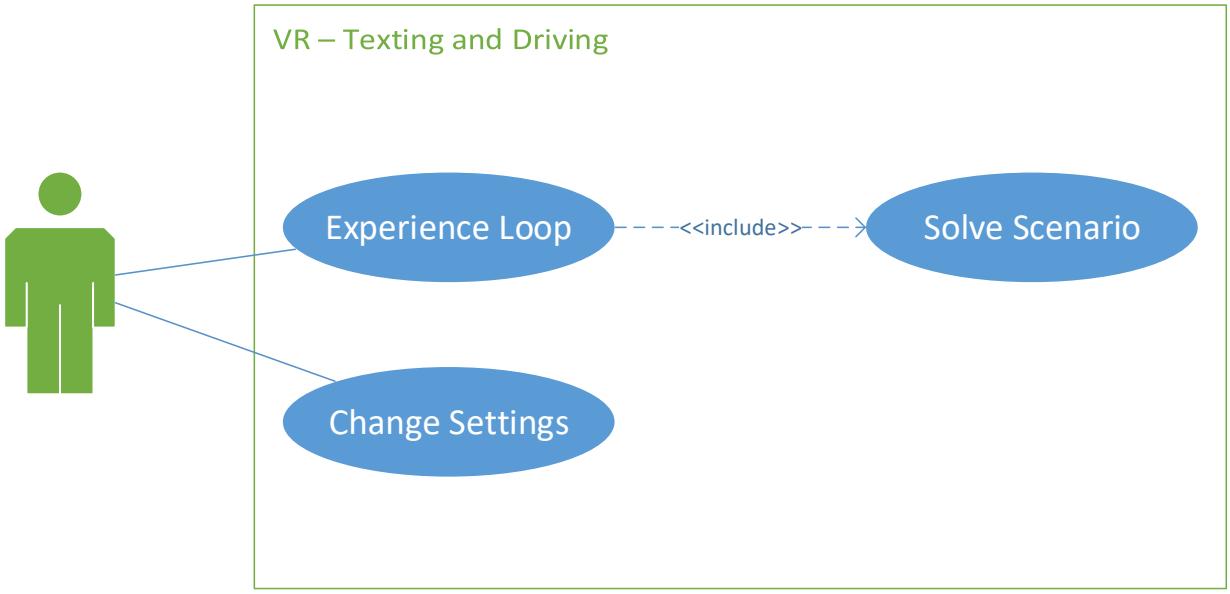

Figure 4.1 - Use Case Diagram

4.1.3.2 User Scenarios

Figure 4.2 lists the details of the use cases that occur within the system. The use cases give an overview of the sequence of the interactions that occur with the user and the system.

Project Name: Virtual Reality---Texting While Driving				
Use Case ID	**Use Case Name**	**Level**	**Author**	**Version**
UC-001	Change Settings	Primary task	Nathan Christiansen	0.4
UC-002	Experience Loop	Primary task	Nathan Christiansen	0.6
UC-003	Solve Scenario	Subfunction	Nathan Christiansen	0.3
Acknowledgment: Generated from the CapStone process management system ©2015				

Figure 4.2 - Use Case List

Project Name:	**Virtual Reality--Texting While Driving**
Use Case ID:	UC-001
Use Case Name:	Change Settings
User Goal:	Change Experience Settings
Scope:	VR - Texting While Driving
Level:	Primary task
Relevant User Reqs:	UF-E
Relevant System Reqs:	SF-E-01
Primary Actor:	User
Precondition:	The application is running and on the main menu
Minimal Guarantee:	Setting changes do not persist
Success Guarantee:	Settings are changed to user specifications
Trigger:	User selects settings option on main menu
Success Scenario:	**Step** **Actions** 1 The user selects settings in the main menu 2 The system brings up the settings menu 3 The user changes their desired settings 4 The user saves changes 5 The system applies changes
Extensions:	**Branching Scenarios**
4A	**Condition:** The user does not save changes
	Step **Actions** 1 The user declines to make changes 2 The system returns to the main menu
Acknowledgment: Generated from the CapStone process management system ©2015	

Figure 4.3 - Change Settings

Project Name:	Virtual Reality---Texting While Driving
Use Case ID:	UC-002
Use Case Name:	Experience Loop
User Goal:	Experience the experience
Scope:	VR - Texting while Driving
Level:	Primary task
Relevant User Reqs:	UF-B,UF-C,UF-D
Relevant System Reqs:	SF-B-01,SF-B-02,SF-C-01,SF-D-01
Primary Actor:	User
Precondition:	The application is running and on the main menu
Minimal Guarantee:	The user enters the experience
Success Guarantee:	The user finishes the experience
Trigger:	User selects start experience on the main menu
Success Scenario:	**Step** **Actions** 1 The user selects start experience on the main menu 2 The system begins the experience 3 The user gains control of the passenger 4 The user SOLVES SCENARIO 5 The system continues until the next threshold 6 The system repeats step 4-5 until the user completes the experience 7 The system displays a results screen to the user
Extensions:	**Branching Scenarios**
5A	**Condition:** The user fails a scenario
	Step **Actions** 1 The system ends the experience
Acknowledgment: Generated from the CapStone process management system ©2015	

Figure 4.4 – Experience Loop

Project Name:	Virtual Reality--Texting While Driving
Use Case ID:	UC-003
Use Case Name:	Solve Scenario
User Goal:	The user makes choices to solve a scenario
Scope:	VR - Texting While Driving
Level:	Subfunction
Relevant User Reqs:	UF-A
Relevant System Reqs:	SF-A-01
Primary Actor:	User
Precondition:	The user is in the experience and has not failed
Minimal Guarantee:	The default solution is chosen
Success Guarantee:	The user's solution is chosen
Trigger:	The user reaches a scenario threshold
Success Scenario:	Step Actions 1 The user reaches a scenario threshold 2 The system presents a scenario involving a dangerous situation 3 The user selects a solution presented by the scenario 4 The system enters a success state for the scenario
Extensions:	Branching Scenarios
3A	Condition: The user selects an incorrect solution or does not enter within the alloted time
	Step Actions 1 The system enters a fail state for the scenario
Acknowledgment: Generated from the CapStone process management system ©2015	

Figure 4.5 - Solve Scenario

4.1.3.3 List of User Functional Requirements

User functional requirements describe functionality that the system should provide.

Project Name:	Virtual Reality--Texting While Driving				
Requirement ID:	UF-A			Type	Functional	Non-Functional
Creation:	Sep 16 2016 12:51 PM			User	☒	☐
Modification:	Sep 30 2016 03:07 PM			System	☐	☐
Description:	The application should present various scenarios that display a distracted driver, and give the user the ability to overcome the potential negative outcome.				
Priority:	✓ Highest	High	Medium	Low	Lowest
This Req. is Refined Into:	SF-A-01				
Justify why UF-A can be completely covered by SF-A-01	SF-A-01 describes how many choices the user will be able to choose from to affect their outcome.				
Traceability:	Use cases cf.	UC-003			
	Test cases cf.	Yet to be completed in test case worksheet!			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Figure 4.6 - Requirement UF-A

Project Name:	Virtual Reality--Texting While Driving				
Requirement ID:	UF-B			Type	Functional	Non-Functional
Creation:	Sep 16 2016 01:05 PM			User	☒	☐
Modification:	Sep 30 2016 03:06 PM			System	☐	☐
Description:	The user should control a passenger in a vehicle driven by a person engaging in dangerous activities.				
Priority:	✓ Highest	High	Medium	Low	Lowest
This Req. is Refined Into:	SF-B-01, SF-B-02				
Justify why UF-B can be completely covered by SF-B-01, SF-B-02	SF-B-01 specifies how the user will be able to control a passenger. SF-B-02 specifies how the user will be able to input commands.				
Traceability:	Use cases cf.	UC-002			
	Test cases cf.	Yet to be completed in test case worksheet!			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Figure 4.7 - Requirement UF-B

Project Name:	Virtual Reality---Texting While Driving				
Requirement ID:	UF-C			Type	Functional	Non-Functional
Creation:	Sep 21 2016 02:59 PM			User	☒	☐
Modification:	Oct 18 2016 08:25 AM			System	☐	☐
Description:	The system should feature multiple outcomes that can occur due to the driver being distracted.				
Priority:	✓ Highest	High	Medium	Low	Lowest
This Req. is Refined Into:	SF-C-01				
Justify why UF-C can be completely covered by SF-C-01	SF-C-01 specifies how many outcomes the system will provide and gives detail about each.				
Traceability:	Use cases cf.	UC-002			
	Test cases cf.	Yet to be completed in test case worksheet!			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Figure 4.8 - Requirement UF-C

Project Name:	Virtual Reality--Texting While Driving				
Requirement ID:	UF-D			Type	Functional	Non-Functional
Creation:	Sep 21 2016 03:00 PM			User	☒	☐
Modification:	Sep 30 2016 03:01 PM			System	☐	☐
Description:	The user should be able to interact with their environment between scenarios presented to them				
Priority:	✓ Highest	High	Medium	Low	Lowest
This Req. is Refined Into:	SF-D-01				
Justify why UF-D can be completely covered by SF-D-01	SF-D-01 specifies the objects that the user will be able to interact with.				
Traceability:	Use cases cf.	UC-002			
	Test cases cf.	Yet to be completed in test case worksheet!			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Figure 4.9 - Requirement UF-D

Project Name:	Virtual Reality--Texting While Driving				
Requirement ID:	UF-E			Type	Functional	Non-Functional
Creation:	Sep 26 2016 03:11 PM			User	☒	☐
Modification:	Sep 30 2016 03:01 PM			System	☐	☐
Description:	User should be able to modify experience settings				
Priority:	Highest	High	✓ Medium	Low	Lowest
This Req. is Refined Into:	SF-E-01				
Justify why UF-E can be completely covered by SF-E-01	SF-E-01 provides some settings that the user can modify to alter the experience.				
Traceability:	Use cases cf.	UC-001			
	Test cases cf.	Yet to be completed in test case worksheet!			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Figure 4.10 - Requirement UF-E

Project Name:	Virtual Reality---Texting While Driving				
Requirement ID:	UF-F			Type / Functional / Non-Functional	
Creation:	Oct 21 2016 12:12 PM			User ☒ / ☐	
Modification:	Oct 21 2016 12:17 PM			System ☐ / ☐	
Description:	The driver should be controlled by an AI and should engage in various tasks.				
Priority:	Highest	✓ High	Medium	Low	Lowest
This Req. is Refined Into:	SF-F-01				
Justify why UF-F can be completely covered by SF-F-01	Specifies what the AI will perform during the experience.				
Traceability:	Use cases cf.	Yet to be completed in use case worksheet!			
	Test cases cf.	Yet to be completed in test case worksheet!			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Figure 4.11 - Requirement UF-F

4.1.4 Non-functional Requirements

Non-functional requirements describe the constraints and quality of the functionalities, providing testable features and specifying restrictions.

4.1.4.1 Product: Usability Requirements

Usability requirements describe how easily a user interacts with the system.

4.1.4.2 Product: Performance Requirements

Performance requirements describe how well a system performs in terms of time and resource usage.

Project Name:	Virtual Reality--Texting While Driving				
Requirement ID:	UP-01			Type / Functional / Non-Functional	
Creation:	Sep 26 2016 03:06 PM			User ☐ / ☒	
Modification:	Sep 30 2016 03:02 PM			System ☐ / ☐	
Description:	The system should run at an acceptable frame rate suitable for virtual reality use.			Product (sub-type below) / Performance Requirements	
Priority:	Highest	High	✓ Medium	Low	Lowest
This Req. is Refined Into:	SP-01-01				
Justify why UP-01 can be completely covered by SP-01-01	Specifies what the acceptable frame rate the system should run at.				
Traceability:	Use cases cf.	N/A			
	Test cases cf.	Yet to be completed in test case worksheet!			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Figure 4.12 - Requirement UP-01

4.1.4.3 Product: Dependability/Security Requirements

Dependability/Security requirements describe the reliability and security concerns of the project.

4.1.4.4 Organizational: Development Requirements

Development requirements specify development practices and constraints.

Project Name:	Virtual Reality--Texting While Driving				
Requirement ID:	UO-01			Type	Functional	Non-Functional
Creation:	Sep 16 2016 12:56 PM			User	☐	☒
Modification:	Sep 30 2016 03:03 PM			System	☐	☐
Description:	The application should be developed for modern Android devices.			Organizational (sub-type below)		
				Development Requirements		
Priority:	✓ Highest	High	Medium	Low	Lowest	
This Req. is Refined Into:	SO-01-01					
Justify why UO-01 can be completely covered by SO-01-01	SO-01-01 specifies a modern Android operating system and device for eventual app deployment.					
Traceability:	Use cases cf.	N/A				
	Test cases cf.	Yet to be completed in test case worksheet!				
Acknowledgment	*Generated from the CapStone Process Management System ©2015*					

Figure 4.13 - Requirement UO-01

Project Name:	Virtual Reality--Texting While Driving				
Requirement ID:	UO-02			Type	Functional	Non-Functional
Creation:	Sep 16 2016 12:58 PM			User	☐	☒
Modification:	Sep 30 2016 03:01 PM			System	☐	☐
Description:	The application should be developed for cardboard VR use.			Organizational (sub-type below)		
				Development Requirements		
Priority:	✓ Highest	High	Medium	Low	Lowest	
This Req. is Refined Into:	SO-02-01					
Justify why UO-02 can be completely covered by SO-02-01	Specifies the SDK and method of displaying virtual reality applications.					
Traceability:	Use cases cf.	N/A				
	Test cases cf.	Yet to be completed in test case worksheet!				
Acknowledgment	*Generated from the CapStone Process Management System ©2015*					

Figure 4.14 - Requirement UO-02

Project Name:	Virtual Reality--Texting While Driving				
Requirement ID:	UO-03			Type	Functional	Non-Functional
Creation:	Sep 16 2016 01:04 PM			User	☐	☒
Modification:	Sep 30 2016 03:01 PM			System	☐	☐
Description:	The application must feature ERIE Insurance branded paraphernalia advertising the company throughout.			Organizational (sub-type below)	
				Development Requirements	
Priority:	Highest	✓ High	Medium	Low	Lowest
This Req. is Refined Into:	SO-03-01				
Justify why UO-03 can be completely covered by SO-03-01	Specifies objects to be textured with ERIE Insurance textures.				
Traceability:	Use cases cf.	N/A			
	Test cases cf.	Yet to be completed in test case worksheet!			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Figure 4.15 - Requirement UO-03

4.1.4.5 Organizational: Operational Requirements

Operational requirements describe conditions that a system must support.

4.1.4.6 Organizational: Environmental Requirements

Environmental requirements describe the look and feel of the system's interface.

4.1.4.7 External: Safety/Security Requirements

Safety/Security requirements detail how the system will interact with other systems, and the security concerns of these interactions.

4.1.4.8 External: Cultural and Social Requirements

Cultural and social requirements describe how the system conforms to cultural and social expectations.

4.1.4.9 External: Political Requirements

Political requirements detail how the system will influence different sections of the company.

4.2 System Requirements

User requirements tend to be vague, so they are refined into system requirements. System requirements engineer and refine the user requirements into many detailed requirements that are much more descriptive and implementable.

4.2.1 Functional Requirements

4.2.1.1 List of System Functional Requirements

Project Name:	Virtual Reality--Texting While Driving				
Requirement ID:	SF-A-01			Type / Functional / Non-Functional	
Creation:	Sep 23 2016 01:00 PM			User ☐ / ☐	
Modification:	Sep 23 2016 01:02 PM			System ☒ / ☐	
Description:	The system should provide three possible solutions for every decision presented.				
Priority:	Highest	High	Medium	Low	Lowest
This Req. is Engineered From:	UF-A				
Justify why meeting SF-A-01 can contribute to the fulfilment of UF-A	Specifies how the user can overcome each situation.				
Traceability:	Use cases cf.	UC-003			
	Test cases cf.	Yet to be completed in test case worksheet!			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Figure 4.16 - Requirement SF-A-01

Project Name:	Virtual Reality---Texting While Driving				
Requirement ID:	SF-A-02			Type / Functional / Non-Functional	
Creation:	Nov 09 2016 02:24 PM			User ☐ / ☐	
Modification:	Nov 09 2016 02:25 PM			System ☒ / ☐	
Description:	When the car passes a trigger, a scenario should be presented				
Priority:	✓ Highest	High	Medium	Low	Lowest
This Req. is Engineered From:	UF-A				
Justify why meeting SF-A-02 can contribute to the fulfilment of UF-A	Handles the beginning of individual scenarios				
Traceability:	Use cases cf.	Yet to be completed in use case worksheet!			
	Test cases cf.	Yet to be completed in test case worksheet!			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Figure 4.17 - Requirement SF-A-02

Project Name:	Virtual Reality--Texting While Driving				
Requirement ID:	SF-B-01			Type	Functional	Non-Functional
Creation:	Sep 23 2016 12:54 PM			User	☐	☐
Modification:	Sep 27 2016 10:55 AM			System	☒	☐
Description:	The user should have a first person perspective during the experience, and can use motion inputs to position the camera.				
Priority:	✓ Highest	High	Medium	Low	Lowest
This Req. is Engineered From:	UF-B				
Justify why meeting SF-B-01 can contribute to the fulfilment of UF-B	Specifies the inputs the user can use to control the passenger.				
Traceability:	Use cases cf.	UC-002			
	Test cases cf.	Yet to be completed in test case worksheet!			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Figure 4.18 - Requirement SF-B-01

Project Name:	Virtual Reality--Texting While Driving				
Requirement ID:	SF-B-02			Type	Functional	Non-Functional
Creation:	Sep 27 2016 10:55 AM			User	☐	☐
Modification:	Sep 27 2016 10:56 AM			System	☒	☐
Description:	The user will use the button on the cardboard headset to interact with objects in the environment, and select choices during scenarios				
Priority:	✓ Highest	High	Medium	Low	Lowest
This Req. is Engineered From:	UF-B				
Justify why meeting SF-B-02 can contribute to the fulfilment of UF-B	Specifies input the user has during control				
Traceability:	Use cases cf.	UC-002			
	Test cases cf.	Yet to be completed in test case worksheet!			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Figure 4.19 - Requirement SF-B-02

Project Name:	Virtual Reality---Texting While Driving				
Requirement ID:	SF-B-03			Type / Functional / Non-Functional	
Creation:	Nov 09 2016 01:58 PM			User ☐ / ☐	
Modification:	Nov 09 2016 02:00 PM			System ☒ / ☐	
Description:	Double clicking the input button will re-center the camera to the front of the car				
Priority:	Highest	✓ High	Medium	Low	Lowest
This Req. is Engineered From:	UF-B				
Justify why meeting SF-B-03 can contribute to the fulfilment of UF-B	This requirement aids the user in their control of the passenger as it allows an easy way to refocus to the front of the car				
Traceability:	Use cases cf.	Yet to be completed in use case worksheet!			
	Test cases cf.	TC-001			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Figure 4.20 - Requirement SF-B-02

Project Name:	Virtual Reality---Texting While Driving				
Requirement ID:	SF-C-01			Type / Functional / Non-Functional	
Creation:	Sep 23 2016 12:59 PM			User ☐ / ☐	
Modification:	Oct 18 2016 08:25 AM			System ☒ / ☐	
Description:	The system should have four types of outcomes that can occur within the environment, including hitting an object, running off the road/lanes, speeding/slowing down, and missing traffic lights.				
Priority:	Highest	✓ High	Medium	Low	Lowest
This Req. is Engineered From:	UF-C				
Justify why meeting SF-C-01 can contribute to the fulfilment of UF-C	Specifies the different situations and outcomes that the user is presented with.				
Traceability:	Use cases cf.	UC-002			
	Test cases cf.	Yet to be completed in test case worksheet!			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Figure 4.21 - Requirement SF-C-01

Project Name:	Virtual Reality--Texting While Driving				
Requirement ID:	SF-D-01			Type	Functional	Non-Functional
Creation:	Sep 23 2016 01:02 PM			User	☐	☐
Modification:	Sep 26 2016 02:56 PM			System	☒	☐
Description:	The user should be able to open/close glove box, interact with objects in the glove box, drink a drink in the cup holder, open/close the window, and adjust the radio.				
Priority:	Highest	✓ High	Medium	Low	Lowest
This Req. is Engineered From:	UF-D				
Justify why meeting SF-D-01 can contribute to the fulfilment of UF-D	It specifies the objects that the user can interact with				
Traceability:	Use cases cf.	UC-002			
	Test cases cf.	Yet to be completed in test case worksheet!			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Figure 4.22 - Requirement SF-D-01

Project Name:	Virtual Reality--Texting While Driving				
Requirement ID:	SF-E-01			Type	Functional	Non-Functional
Creation:	Sep 30 2016 01:07 PM			User	☐	☐
Modification:	Sep 30 2016 01:08 PM			System	☒	☐
Description:	The system will provide options to the user including changing weather effects and time of day.				
Priority:	Highest	✓ High	Medium	Low	Lowest
This Req. is Engineered From:	UF-E				
Justify why meeting SF-E-01 can contribute to the fulfilment of UF-E	The user will be given some control of the environment that they participate in.				
Traceability:	Use cases cf.	UC-001			
	Test cases cf.	Yet to be completed in test case worksheet!			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Figure 4.23 - Requirement SF-E-01

Project Name:	Virtual Reality---Texting While Driving				
Requirement ID:	SF-F-01				
Creation:	Oct 21 2016 12:13 PM				
Modification:	Oct 21 2016 12:16 PM				
Description:	The driver AI should drive, text, converse with user, and look out window.				
Priority:	Highest	✓ High	Medium	Low	Lowest
This Req. is Engineered From:	UF-F				
Justify why meeting SF-F-01 can contribute to the fulfilment of UF-F	Specifies exactly what the driver's AI will do during the experience.				
Traceability:	Use cases cf.	Yet to be completed in use case worksheet!			
	Test cases cf.	Yet to be completed in test case worksheet!			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Type	Functional	Non-Functional
User	☐	☐
System	☒	☐

Figure 4.24 - Requirement SF-F-01

4.2.1.2 System Behavior

Figures 4.23 and 4.24 detail the sequence of flow between user and system, much like use cases. However, they give a more detailed look into the system, providing interaction between components in the system as well.

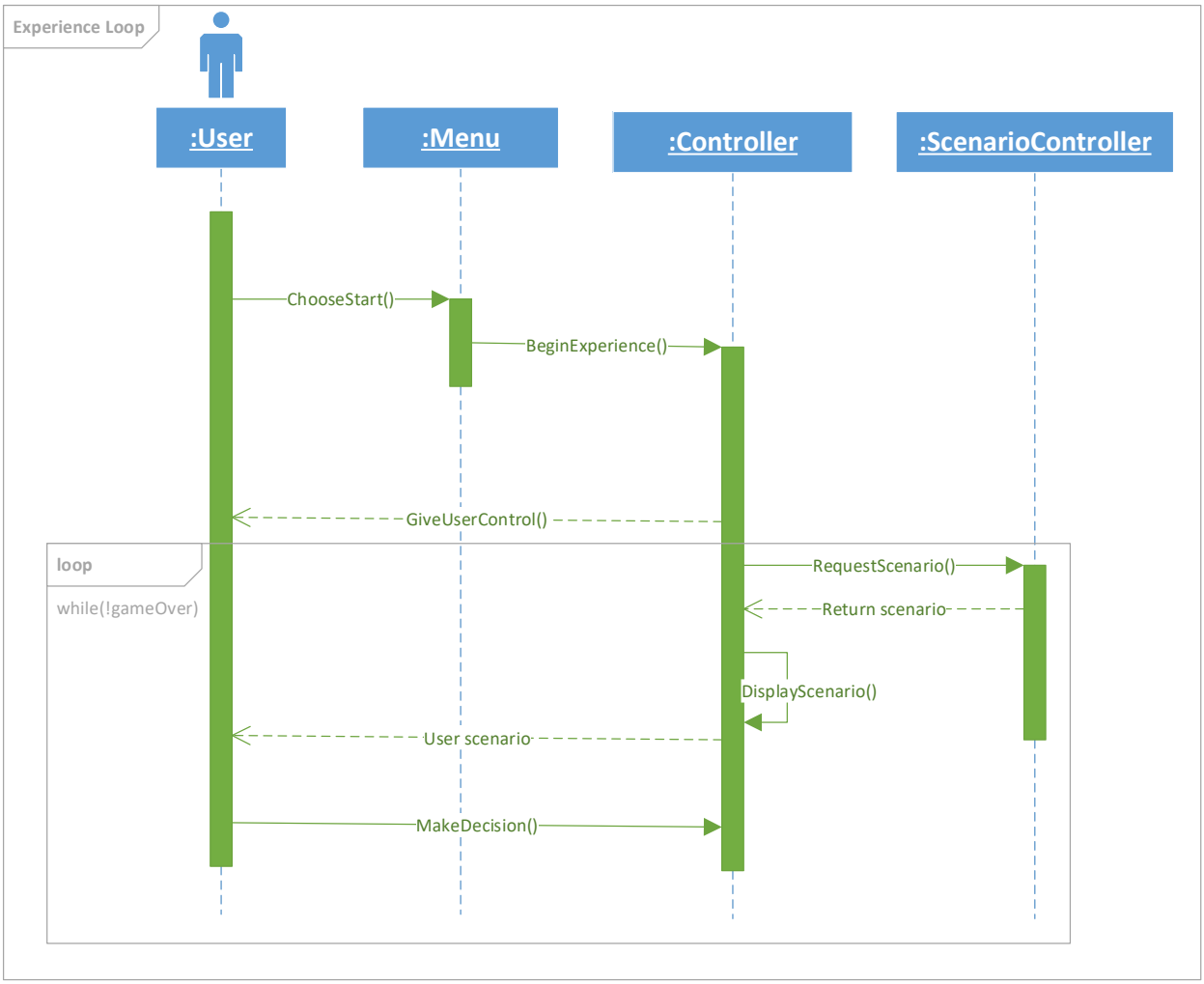

Figure 4.25 - Experience Loop Sequence

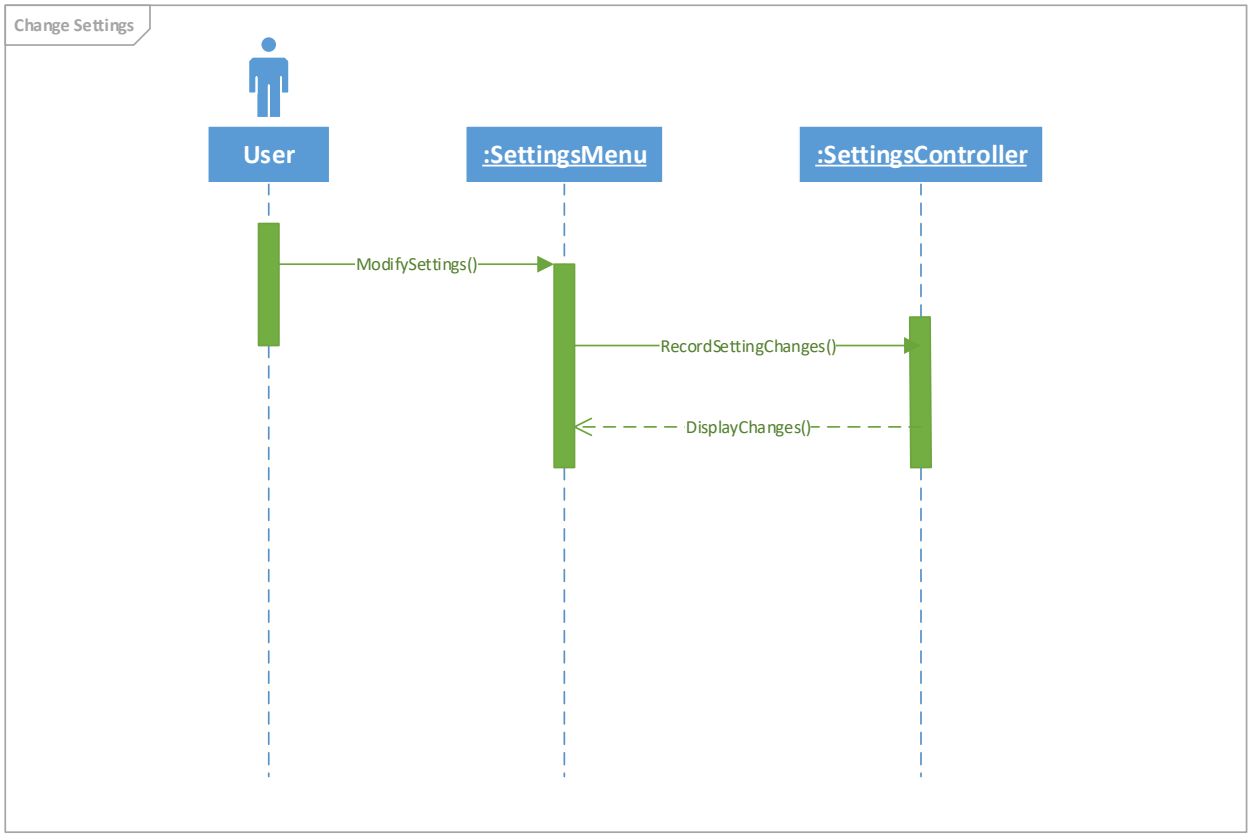

Figure 4.26 - Change Settings Sequence

4.2.1.3 Data Requirements

4.2.2 Non-functional Requirements

4.2.2.1 Product: Usability Requirements

4.2.2.2 Product: Performance Requirements

Project Name:	Virtual Reality--Texting While Driving				
Requirement ID:	SP-01-01			Type / Functional / Non-Functional	
Creation:	Sep 30 2016 02:54 PM			User ☐ / ☐	
Modification:	Sep 30 2016 02:55 PM			System ☐ / ☒	
Description:	The application should run at a minimum of 30 frames per second.			Product (sub-type below)	
				Performance Requirements	
Priority:	Highest	✓ High	Medium	Low	Lowest
This Req. is Engineered From:	UP-01				
Justify why meeting SP-01-01 can contribute to the fulfilment of UP-01	Specifies the minimum fps that the experience should perform at.				
Traceability:	Use cases cf.	N/A			
	Test cases cf.	Yet to be completed in test case worksheet!			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Figure 4.27 - Requirement SP-01-01

4.2.2.3 Product: Dependability/Security Requirements

4.2.2.4 Organizational: Development Requirements

Project Name:	Virtual Reality--Texting While Driving				
Requirement ID:	SO-01-01			Type / Functional / Non-Functional	
Creation:	Sep 26 2016 02:59 PM			User ☐ / ☐	
Modification:	Sep 26 2016 03:02 PM			System ☐ / ☒	
Description:	The system should be targeted for Android 5.1.1 "Lollipop" for phones with hardware specifications of the Samsung S5 and up			Organizational (sub-type below)	
				Development Requirements	
Priority:	Highest	✓ High	Medium	Low	Lowest
This Req. is Engineered From:	UO-01				
Justify why meeting SO-01-01 can contribute to the fulfilment of UO-01	Specifies the OS version and hardware requirements				
Traceability:	Use cases cf.	N/A			
	Test cases cf.	Yet to be completed in test case worksheet!			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Figure 4.28 - Requirement SO-01-01

Project Name:	Virtual Reality--Texting While Driving				
Requirement ID:	SO-02-01			Type	Functional	Non-Functional
Creation:	Sep 30 2016 01:03 PM			User	☐	☐
Modification:	Sep 30 2016 01:05 PM			System	☐	☒
Description:	The system will utilize the Google VR SDK to display two images through the cardboard.			Organizational (sub-type below)		
				Development Requirements		
Priority:	✓ Highest	High	Medium	Low	Lowest	
This Req. is Engineered From:	UO-02				
Justify why meeting SO-02-01 can contribute to the fulfilment of UO-02	The system will provide a VR experience that is designed around the cardboard.				
Traceability:	Use cases cf.	N/A			
	Test cases cf.	Yet to be completed in test case worksheet!			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Figure 4.29 - Requirement SO-02-01

Project Name:	Virtual Reality--Texting While Driving				
Requirement ID:	SO-03-01			Type	Functional	Non-Functional
Creation:	Sep 30 2016 01:00 PM			User	☐	☐
Modification:	Sep 30 2016 01:02 PM			System	☐	☒
Description:	Erie Insurance logos will be placed on buildings, billboards, bumper stickers, and air fresheners.			Organizational (sub-type below)		
				Development Requirements		
Priority:	Highest	High	✓ Medium	Low	Lowest	
This Req. is Engineered From:	UO-03				
Justify why meeting SO-03-01 can contribute to the fulfilment of UO-03	Erie Insurance will be represented within the experience.				
Traceability:	Use cases cf.	N/A			
	Test cases cf.	Yet to be completed in test case worksheet!			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Figure 4.30 - Requirement SO-03-01

4.2.2.5 Organizational: Operational Requirements

4.2.2.6 Organizational: Environmental Requirements

4.2.2.7 External: Safety/Security Requirements

4.2.2.8 External: Cultural and Social Requirements

4.2.2.9 External: Political Requirements

4.3 Requirements Trace Table

Figure 4.29 gives a breakdown of the system requirements that have been engineered from the user requirements.

Project Name: Virtual Reality---Texting While Driving			
User Requirements		**System Requirements**	
Req ID	**Description**	**Req ID**	**Description**
UF-A	The application should present various scenarios that display a distracted driver, and give the user the ability to overcome the potential negative outcome.	SF-A-01	The system should provide three possible solutions for every decision presented.
UF-B	The user should control a passenger in a vehicle driven by a person engaging in dangerous activities.	SF-B-01	The user should have a first person perspective during the experience, and can use motion inputs to position the camera.
		SF-B-02	The user will use the button on the cardboard headset to interact with objects in the environment, and select choices during scenarios
UF-C	The system should feature multiple outcomes that can occur due to the driver being distracted.	SF-C-01	The system should have four types of outcomes that can occur within the environment, including hitting an object, running off the road/lanes, speeding/slowing down, and missing traffic lights.
UF-D	The user should be able to interact with their environment between scenarios presented to them	SF-D-01	The user should be able to open/close glove box, interact with objects in the glove box, drink a drink in the cup holder, open/close the window, and adjust the radio.
UF-E	User should be able to modify experience settings	SF-E-01	The system will provide options to the user including changing weather effects and time of day.
UF-F	The driver should be controlled by an AI and should engage in various tasks.	SF-F-01	The driver AI should drive, text, converse with user, and look out window.
UO-01	The application should be developed for modern Android devices.	SO-01-01	The system should be targeted for Android 5.1.1 "Lollipop" for phones with hardware specifications of the Samsung S5 and up
UO-02	The application should be developed for cardboard VR use.	SO-02-01	The system will utilize the Google VR SDK to display two images through the cardboard.
UO-03	The application must feature ERIE Insurance branded paraphernalia advertising the company throughout.	SO-03-01	Erie Insurance logos will be placed on buildings, billboards, bumper stickers, and air fresheners.
UP-01	The system should run at an acceptable frame rate suitable for virtual reality use.	SP-01-01	The application should run at a minimum of 30 frames per second.
Acknowledgment: Generated from the CapStone process management system ©2015			

Figure 4.31 - Requirement Trace Table

5. Exploratory Studies

5.1 Relevant Techniques

We will be using the Unity 3D game engine to create our application. We have chosen this engine because of its C# scripting, large community, and because it allows us to create an immersive VR experience very quickly. Along with Unity 3D, we will be using the Google VR SDK for Unity to adapt our project for VR use [6]. We also plan to take advantage of the Unity Asset Store to collect models, animations, and scripts to allow us to focus on implementing the requested features and not worry about having to create all of our assets from scratch. Within the Asset Store exists an important package called Unity Test Tools [4]. Unity Test Tools allows us various ways of testing including unit tests, integration tests, and assertion component to make sure our work is as bug free as possible. All of these technologies working together will allow us to create an experience that puts the user into the middle of a seemingly dangerous situation.

5.2 Relevant Packages/Products

The main products and packages we will be using include Unity 3D, Google VR SDK, a variety of assets from the Unity Asset Store, the Android SDK to build from within the Unity engine, Unity Test Tools to complete our application testing, Visual Studio for writing C# scripts, and potentially more as we move forward.

5.3 Broader Impacts

This virtual reality experience has the potential to help minimize distracted driving. Minimizing distracted driving means that there will be less accidents, less injuries, and less deaths because of distracted driving. Since the application runs on the Android operating system, which is used by millions of people every day, this application has the potential to reach a large number of drivers and passengers.

6. System Design

6.1 Architectural Design

The system will be using a component-based architectural design, which emphasizes the creation of components, which other components reuse to create a scene. Multiple scenes are sequenced together to create the overall system. Unity objects are a component that is self-contained, meaning that it can run on its own inside a scene. As objects are defined, they can be used in other objects to create large components that are combined to create complex scenes. Figure 6.1 shows our high-level architecture, which is consisting of a starting interface *GameObject* that has a composition with itself to allow the components to have other components that make it up.

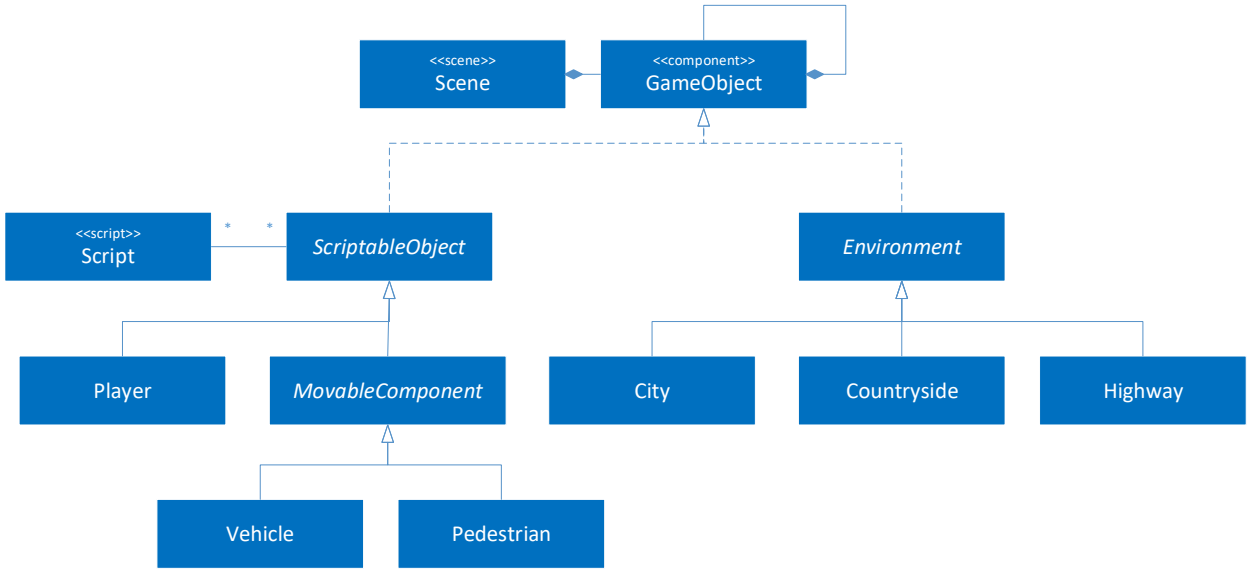

Figure 6.1 - Architectural Design

6.2 Structural Design

The structural diagram provides the detailed components that are defined in the architectural design. The basic components are refined into each individual component that can be reused to create the overall layout of the Unity scene.

Figure 6.2 represents the Scenes package within the structural diagram. This package will contain each scene within the experience and show how they connect to each other. This package also contains the main GameObject interface that all other components will be inheriting from throughout the system.

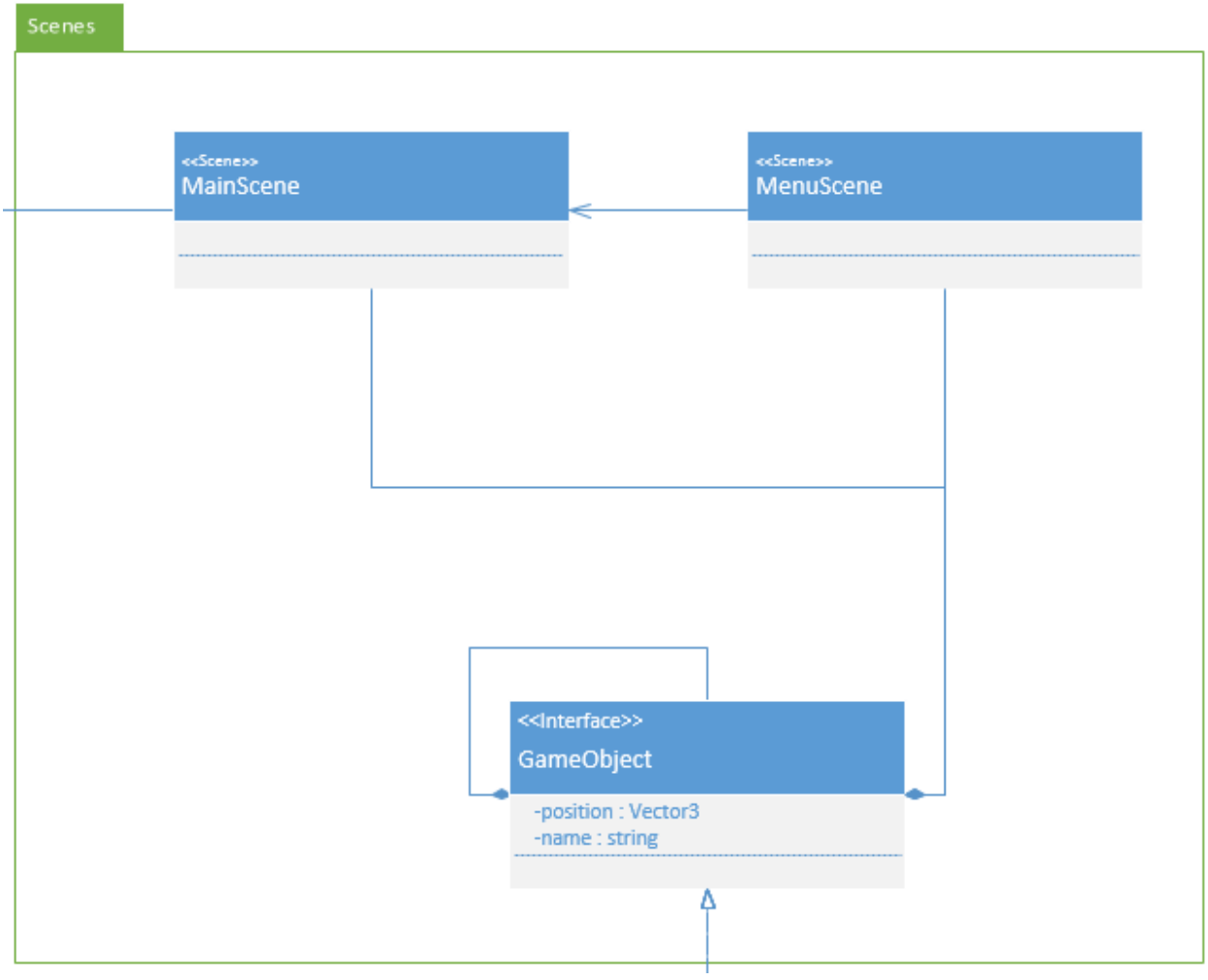

Figure 6.2 - Structural Diagram (Scenes)

Figure 6.3 is the package representing the player. The player is essentially a camera component using a player controller script to allow the user to move their head around to view and interact with what is happening in the scene. The player controller is able to perform actions with the environment such as reentering the user, which is shown below.

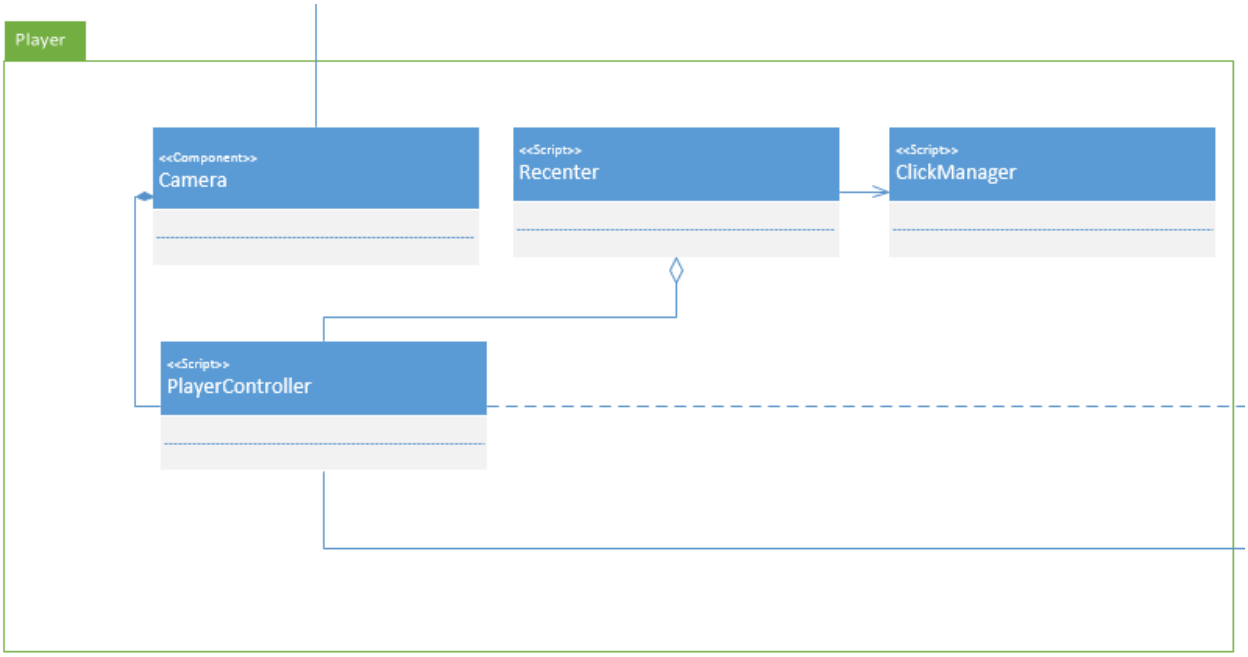

Figure 6.3 - Structural Diagram (Player)

Figure 6.4 shows many of the components that come from Googles VR SDK. As mentioned above, the player is a camera that is able to interact with the environment. To do this, the camera utilizes components, interfaces, and scripts in this package. This package allows components to be set as either objects causing interactions to happen or allows components to be the object that is interacted with.

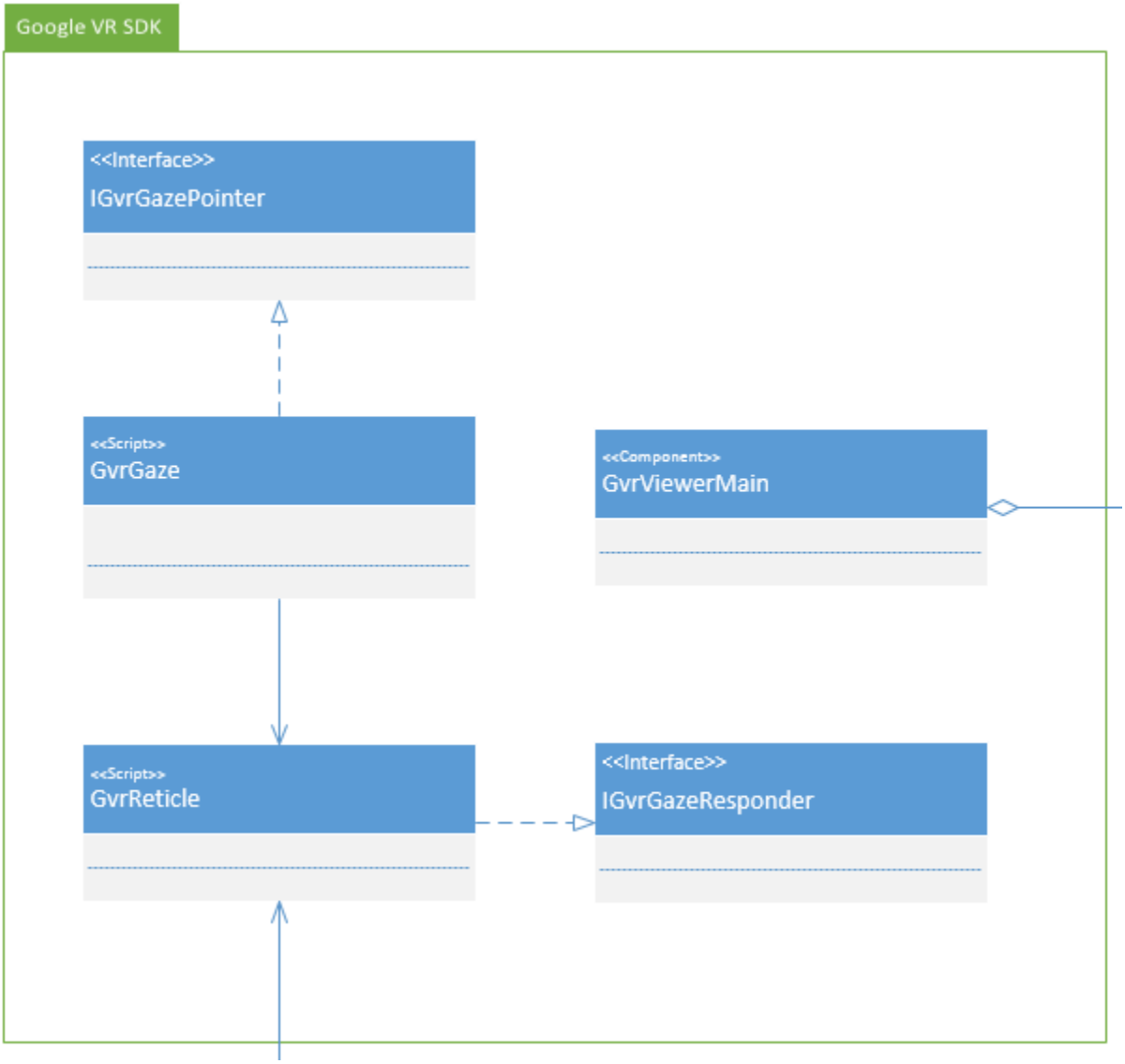

Figure 6.4 - Structural Diagram (Google VR SDK)

Figure 6.5 shows the MovableComponent package which consists of all components that will be moving in some way during the execution of the program. This package includes pedestrians (people, animals, etc.) and vehicles. The package also contains the scripts that these components will rely on to perform their movement and coordination.

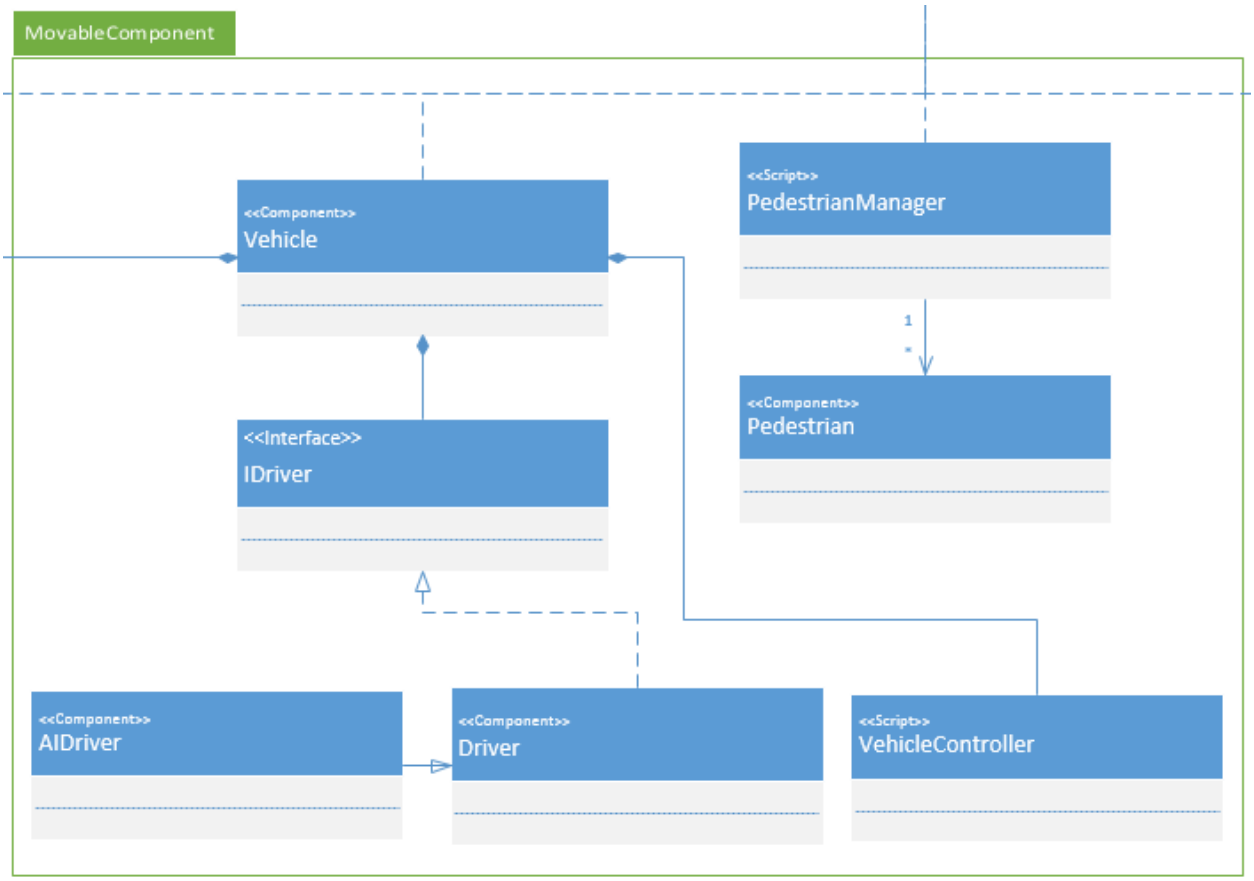

Figure 6.5 - Structural Diagram (MovableComponent)

Figure 6.6 shows the EnvironmentalObjects package which contains objects that are non-moving and exist in the environment such as plants, buildings, and roadways. The hierarchy below demonstrates how full environments will be made up of smaller components such as what was listed previously.

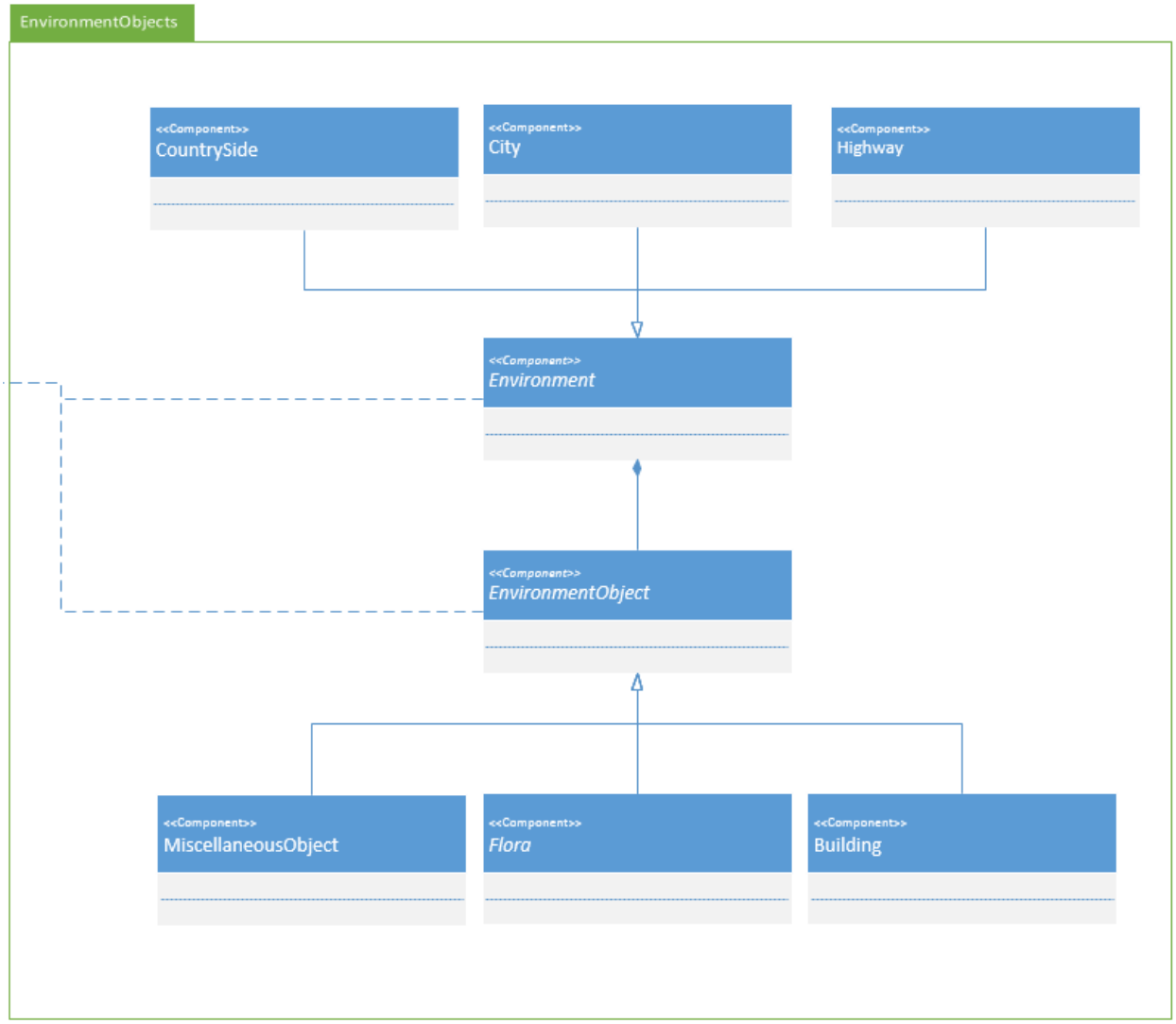

Figure 6.6 - Structural Diagram (EnvironmentObjects)

6.3 User Interface Design

Our user interface design is built around the technologies we are implementing. Virtual reality has a defined structure of displaying an image on two separate screens with logical angles that simulate what eyes see. With that, we are trying to create a very realistic depiction of riding in the car with a friend while the friend engages in dangerous activities. Google's SDK has provided many useful assets that have helped create the menu screens and input management to allow the player to control the experience. Figures 6.7 through 6.9 show the view the player has throughout the experience.

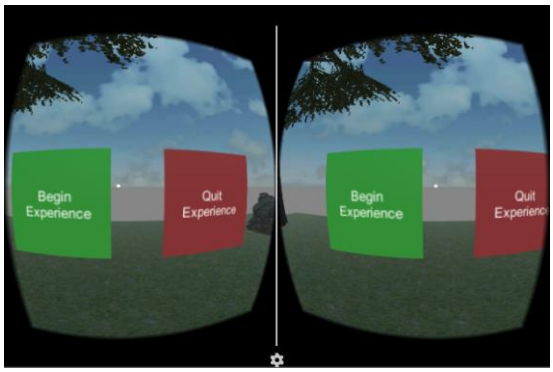

Figure 6.7 - Start Menu View

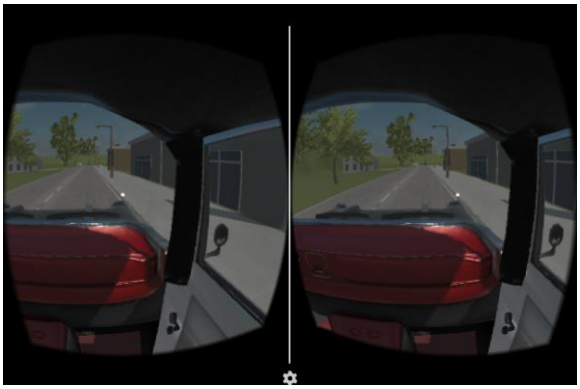

Figure 6.8 - Outside View

Figure 6.9 - Driver View

6.4 Behavioral Design

In Figure 6.10, the behavior of the system is displayed. The activity diagram shows the flow of the experience and gives the steps required to succeed in the system, as well as the fail state requirements.

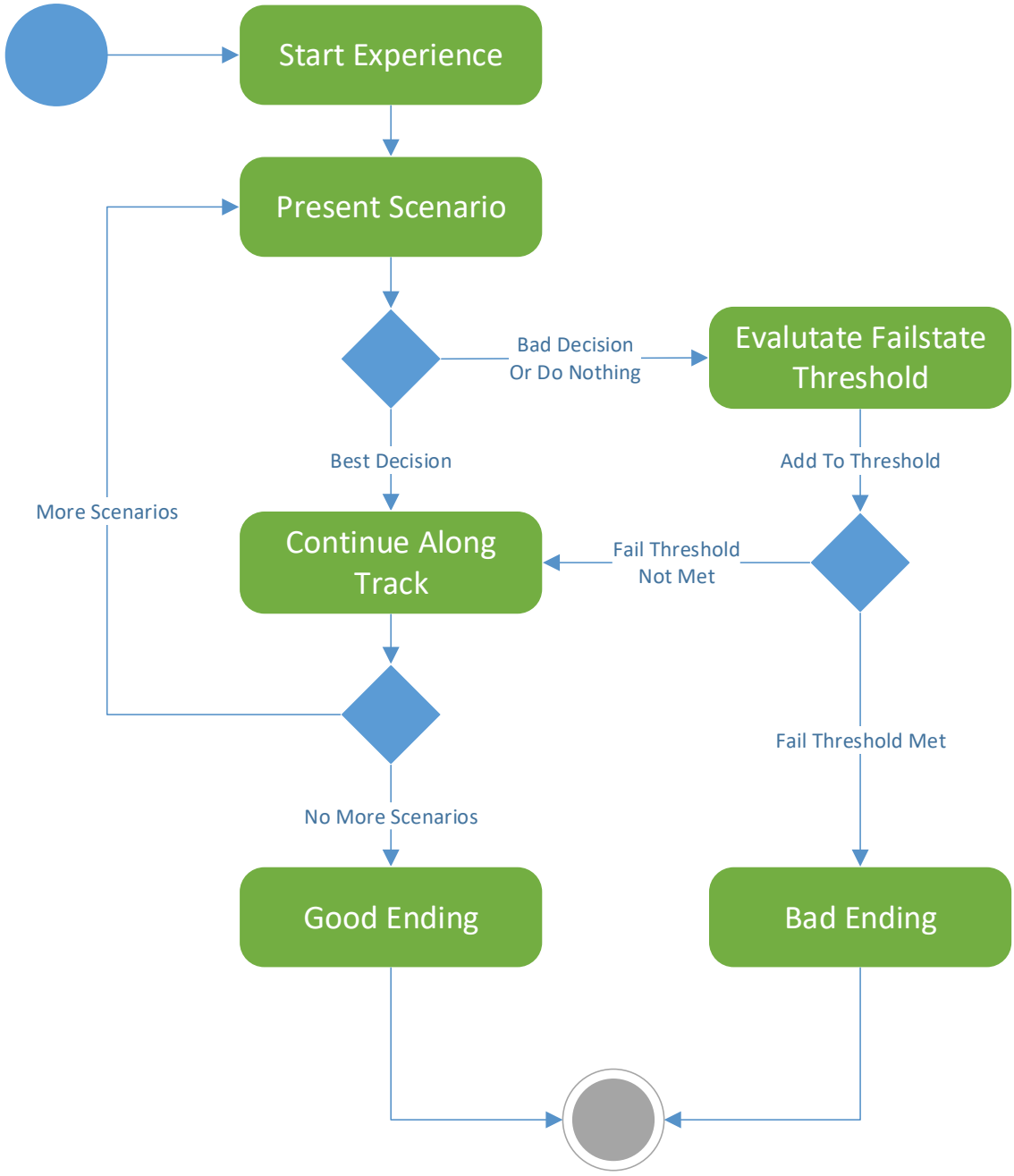

Figure 6.10 - Activity Diagram

6.5 Design Alternatives & Design Rationale

With our project, we are using Unity to create an experience that can run on mobile devices. Unity is designed with the component-based architecture in mind, and the way objects are implemented is based around that concept. Initially we looked into MVC, which is similar to our current design. However, each component in Unity essentially has its own model, view, and controller. The design would be complicated, and would not be as accurate as the component-based architecture.

7. System Implementation

7.1 Programming Languages & Tools

We are implementing our project using Unity, which takes advantage of C# for creating scripts. Unity provides an IDE called MonoDevelop, however we are using Microsoft Visual Studio, which can be used instead of MonoDevelop.

7.2 Coding Conventions

We will adhere to the coding conventions designed around Unity development as well as Microsoft's C# conventions. We will also be following Unity best practice for component design, which will help improve maintainability and performance.

7.3 Code Version Control

As with all projects being worked on by multiple personnel, version control is very important for the efficiency of our workflow. We will be using a combination of Git and Unity SmartMerge for our version control which will handle branching and merge conflicts. We will be hosting our repository in GitHub.

7.4 Implementation Alternatives & Decision Rationale

One alternative development tool we could have used instead of Unity is Unreal Engine. Unreal is another game engine that is widely available and features mobile development and also has Google Cardboard SDK support. With Unreal we would also be developing using C++ instead of C#. Our team decided to use Unity over Unreal because we are all more familiar with C# and virtual reality development is more popular with Unity, so the documentation and resources available will be better defined. Erie Insurance has stated that they are aware of the terms of service with Unity are we are still fine to proceed with development.

7.5 Analysis of Key Algorithms

N/A

8. System Testing

8.1 Test Automation Framework

Our project is developed following the test-driven development methodology. In section 8, we will be covering the tests designed for our application as we continue developing it. In order for our system requirements to be verified, there will be tests created for each one to ensure correct implementation.

8.1.1 Steps for Installing Test Framework

Our tests are designed using Unity Test Tools, which is an asset that allows assertions on Unity objects and scripts to verify that everything is working correctly. In order to install the testing framework, all that needs to be done is download Unity Test Tools from the Unity Asset Store and add it to an existing project.

8.1.2 Steps for Running Test Cases

In order to run a test case, the test case must be opened in Unity Test Tools. From there, the tests can be ran or modified to specified settings.

8.2 Test Case Design

8.2.1 Acceptance Test Cases

These test cases are specifically tailored to test user requirements. The tests verify that specific requirements are working as planned for the user. They ensure that the most important requirements provided by the end user are covered within the system.

Project Name:	Virtual Reality---Texting While Driving	
Test Suite	TS-002: Environment Interaction	
Test Case ID	TC-009 (Acceptance Test)	
What To Test	AI Driver Actions	
Test Data Input		
Expected Result	The AI driver drives and converses with the user throughout the experience, and gets realistically distracted when scenarios are triggered through texting/not paying attention to the road.	
Traceability	Relevant User Req.(s)	UF-F
	Relevant System Req.(s)	SF-F-01
	Relevant Use Case(s)	UC-002
Acknowledgment: Generated from the CapStone process management system ©2015		

Figure 8.1 - AI Driver Actions Test

Project Name:	Virtual Reality---Texting While Driving	
Test Suite	TS-003: System Performance	
Test Case ID	TC-010 (Acceptance Test)	
What To Test	Hardware Validation	
Test Data Input		
Expected Result	The system runs smoothly on hardware specifications of the Samsung S5 and up.	
Traceability	Relevant User Req.(s)	UO-01
	Relevant System Req.(s)	SO-01-01
	Relevant Use Case(s)	
Acknowledgment: Generated from the CapStone process management system ©2015		

Figure 8.2 - Hardware Validation Test

Project Name:	Virtual Reality---Texting While Driving	
Test Suite	TS-003: System Performance	
Test Case ID	TC-011 (Acceptance Test)	
What To Test	Double Image VR Display	
Test Data Input		
Expected Result	Two images should be displayed for use with the Google Cardboard.	
Traceability	Relevant User Req.(s)	UO-02
	Relevant System Req.(s)	SO-02-01
	Relevant Use Case(s)	
Acknowledgment: Generated from the CapStone process management system ©2015		

Figure 8.3 - VR Display Test

Project Name:	Virtual Reality---Texting While Driving	
Test Suite	TS-003: System Performance	
Test Case ID	TC-012 (Acceptance Test)	
What To Test	System Frame Rate	
Test Data Input		
Expected Result	The system runs at or above 30 frames per second when viewed through on a mobile device through a Google Cardboard.	
Traceability	Relevant User Req.(s)	UP-01
	Relevant System Req.(s)	SP-01-01
	Relevant Use Case(s)	
Acknowledgment: Generated from the CapStone process management system ©2015		

Figure 8.4 – System Frame Rate Test

8.2.2 System Test Cases

System tests covers major system functionalities, and tests specific system requirements.

Project Name:	Virtual Reality---Texting While Driving	
Test Suite	TS-001: Scenario Interaction	
Test Case ID	TC-002 (System Test)	
What To Test	Scenario Triggers	
Test Data Input		
Expected Result	A scenario is presented to the user upon reaching a trigger in the environment.	
Traceability	Relevant User Req.(s)	UF-A
	Relevant System Req.(s)	SF-A-02
	Relevant Use Case(s)	UC-002,UC-003
Acknowledgment: Generated from the CapStone process management system ©2015		

Figure 8.5 - Scenario Trigger Test

Project Name:	Virtual Reality---Texting While Driving	
Test Suite	TS-001: Scenario Interaction	
Test Case ID	TC-006 (System Test)	
What To Test	Bad Scenario Outcomes	
Test Data Input		
Expected Result	Choosing the wrong decision or doing nothing at the end of a scenario will sometimes lead to one of the four types of bad outcomes.	
Traceability	Relevant User Req.(s)	UF-C
	Relevant System Req.(s)	SF-C-01
	Relevant Use Case(s)	UC-003
Acknowledgment: Generated from the CapStone process management system ©2015		

Figure 8.6 - Bad Scenario Test

Project Name:	Virtual Reality---Texting While Driving	
Test Suite	TS-002: Environment Interaction	
Test Case ID	TC-008 (System Test)	
What To Test	Weather Modification	
Test Data Input		
Expected Result	If the user changes the weather settings, the environment will reflect those changes.	
Traceability	Relevant User Req.(s)	UF-E
	Relevant System Req.(s)	SF-E-01
	Relevant Use Case(s)	UC-001,UC-002
Acknowledgment: Generated from the CapStone process management system ©2015		

Figure 8.7 - Weather Modification Test

8.2.3 Integration Test Cases

Integration test cases test the connection between the units of a system or subsystem.

Project Name:	Virtual Reality---Texting While Driving	
Test Suite	TS-001: Scenario Interaction	
Test Case ID	TC-003 (Integration Test)	
What To Test	Possible Scenario Solutions	
Test Data Input	None	
Expected Result	3 solutions appear in the user's field of view after a scenario is played out.	
Traceability	Relevant User Req.(s)	UF-A
	Relevant System Req.(s)	SF-A-01
	Relevant Use Case(s)	UC-003
Acknowledgment: Generated from the CapStone process management system ©2015		

Figure 8.8 - Scenario Solutions Test

Project Name:	Virtual Reality---Texting While Driving	
Test Suite	TS-002: Environment Interaction	
Test Case ID	TC-005 (Integration Test)	
What To Test	Environment Interaction	
Test Data Input		
Expected Result	Tapping the input button on environment objects will allow the user to interact with them in some way.	
Traceability	Relevant User Req.(s)	UF-B
	Relevant System Req.(s)	SF-B-02
	Relevant Use Case(s)	UC-002,UC-003
Acknowledgment: Generated from the CapStone process management system ©2015		

Figure 8.9 – Environment Interaction Test

8.2.4 Unit Test Cases

Unit test cases test all parts of an individual unit within a system or subsystem.

Project Name:	Virtual Reality---Texting While Driving	
Test Suite	TS-002: Environment Interaction	
Test Case ID	TC-001 (Unit Test)	
What To Test	Camera Recentering	
Test Data Input		
Expected Result	Camera resets to the default view looking out of the windshield of the vehicle upon two rapid clicks of the input button.	
Traceability	Relevant User Req.(s)	UF-B
	Relevant System Req.(s)	SF-B-03
	Relevant Use Case(s)	UC-002
Acknowledgment: Generated from the CapStone process management system ©2015		

Figure 8.10 – Camera Recentering Test

Project Name:	Virtual Reality---Texting While Driving	
Test Suite	TS-001: Scenario Interaction	
Test Case ID	TC-004 (Unit Test)	
What To Test	Camera Rotation	
Test Data Input		
Expected Result	Rotating the phone moves the camera in the experience uniformly.	
Traceability	Relevant User Req.(s)	UF-B
	Relevant System Req.(s)	SF-B-01
	Relevant Use Case(s)	UC-002
Acknowledgment: Generated from the CapStone process management system ©2015		

Figure 8.11 – Camera Rotation Test

8.3 Test Case Execution Report

The test case execution reports outline the steps taken to execute a given test case. They also provide the status of the test and any defects that will prevent the test from passing.

8.3.1 Unit Testing Report

Project Name:	Virtual Reality---Texting While Driving
Test Case ID:	TC-001
Testing Tools Used:	
Testing Type:	Function coverage
Execution Steps:	1 Begin the experience 2 Turn the camera in some direction away from the default view 3 Quickly double tap the Cardboard input button

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Nick Kapty	11/9/2016	Double tapping does nothing	Fail	Not yet implemented	
2	Nick Kapty	11/15/2016	Double tapping recenters the camera	Pass		

Execution Summary:	Upon implementation, the feature works as intended.

Acknowledgment: Generated from the CapStone process management system ©2015

Figure 8.12 – Camera Recentering Execution

Project Name:	Virtual Reality---Texting While Driving
Test Case ID:	TC-004
Testing Tools Used:	
Testing Type:	Function coverage
Execution Steps:	1 Begin the experience 2 Rotate Cardboard in any direction

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Nick Kapty	11/9/2016	Camera moves around	Pass		

Execution Summary:	The feature works as intended

Acknowledgment: Generated from the CapStone process management system ©2015

Figure 8.13 – Camera Rotating Execution

8.3.2 Integration Testing Report

Project Name:	Virtual Reality---Texting While Driving
Test Case ID:	TC-003
Testing Tools Used:	
Testing Type:	Function coverage
Execution Steps:	1 Begin the experience 2 Wait for the vehicle to reach the first trigger point 3 Wait for the scenario to play out

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Nick Kapty	11/9/2016	No solutions appear	Fail	Not yet implemented	

Execution Summary:	

Acknowledgment: Generated from the CapStone process management system ©2015

Figure 8.14 – Scenario Solutions Execution

Project Name:	Virtual Reality---Texting While Driving
Test Case ID:	TC-005
Testing Tools Used:	
Testing Type:	Function coverage
Execution Steps:	1 Begin the experience 2 Look at an interactive environment object 3 Click on object if reticule expands

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Nick Kapty	11/9/2016	Object does not move	Fail	Not yet implemented	

Execution Summary:	

Acknowledgment: Generated from the CapStone process management system ©2015

Figure 8.15 – Environment Interaction Execution

8.3.3 System Testing Report

Project Name:	Virtual Reality---Texting While Driving	
Test Case ID:	TC-002	
Testing Tools Used:		
Testing Type:	Function coverage	
Execution Steps:	1	Begin the experience
	2	Allow the car to proceed to a predefined trigger point

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Nick Kapty	11/9/2016	No scenario presented	Fail	Not yet implemented	

Execution Summary:	

Acknowledgment: Generated from the CapStone process management system ©2015

Figure 8.16 – Scenario Interaction Execution

Project Name:	Virtual Reality---Texting While Driving	
Test Case ID:	TC-006	
Testing Tools Used:		
Testing Type:	Function coverage	
Execution Steps:	1	Begin the experience
	2	Wait for the vehicle to move to the first scenario
	3	Wait for the scenario to play out
	4	Choose the wrong decision presented

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Nick Kapty	11/9/2016	No outcomes occur	Fail	Not yet implemented	

Execution Summary:	

Acknowledgment: Generated from the CapStone process management system ©2015

Figure 8.17 – Bad Outcomes Execution

Project Name:	Virtual Reality---Texting While Driving	
Test Case ID:	TC-008	
Testing Tools Used:		
Testing Type:	Function coverage	
Execution Steps:	1	Choose the change settings menu
	2	Under the weather tab, select any alternate weather effect
	3	Exit the change settings menu
	4	Begin the experience

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Nick Kapty	11/9/2016	Weather does not change	Fail	Not yet implemented	

Execution Summary:	

Acknowledgment: Generated from the CapStone process management system ©2015

Figure 8.18 – Weather Modification Execution

8.3.4 Acceptance Testing Report

Project Name:	Virtual Reality---Texting While Driving	
Test Case ID:	TC-009	
Testing Tools Used:		
Testing Type:	Function coverage	
Execution Steps:	1	Begin the experience
	2	Look at/Listen to the driver before scenario for animation/conversation
	3	Look at/Listen to the driver during scenario for animation/conversation

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Nick Kapty	11/9/2016	Driver does not animate or interact with the user in any way	Fail		

Execution Summary:	

Acknowledgment: Generated from the CapStone process management system ©2015

Figure 8.19 – AI Driver Interaction Execution

Project Name:	Virtual Reality---Texting While Driving					
Test Case ID:	TC-010					
Testing Tools Used:						
Testing Type:	Function coverage					
Execution Steps:	1	Build the application in Unity				
	2	Export to and attempt to launch the app on an Android phone				
	3	Begin the experience				
Test Execution Records:						
#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Nick Kapty	11/9/2016	App launches successfully	Pass		
Execution Summary:	The app was able to launch on a phone of comparable hardware to the Samsung S5 successfully.					
Acknowledgment: Generated from the CapStone process management system ©2015						

Figure 8.20 – Hardware Validation Execution

Project Name:	Virtual Reality---Texting While Driving					
Test Case ID:	TC-011					
Testing Tools Used:						
Testing Type:	Function coverage					
Execution Steps:	1	Export built app to an Android phone				
	2	Launch app				
Test Execution Records:						
#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Nick Kapty	11/9/2016	The app displayed with a binocular view	Pass		
Execution Summary:	The app ran correctly with a binocular view using the Google VR SDK.					
Acknowledgment: Generated from the CapStone process management system ©2015						

Figure 8.21 – VR Display Execution

Project Name:	Virtual Reality---Texting While Driving
Test Case ID:	TC-012
Testing Tools Used:	
Testing Type:	Function coverage
Execution Steps:	1 Export the built app to an Android phone 2 Launch the app 3 Begin the experience 4 Monitor the FPS throughout the experience using the debug menu

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Nick Kapty	11/9/2016	FPS unknown	Fail	Not yet implemented	

Execution Summary:	

Acknowledgment: Generated from the CapStone process management system ©2015

Figure 8.22 – System Frame Rate Execution

9. Challenges & Open Issues

9.1 Challenges Faced in Requirements Engineering

We had trouble dealing with somewhat vague requirements provided by the industry sponsor, and were faced with the task of continuous meetings in order to get a clear understanding of the sponsor's needs in regard to the system.

9.2 Challenges Faced in System Development

Our first issue we faced was configuring version control to work with our system. Git alone does not work for Unity projects, and scenes are stored in binary files, so if a scene was worked on concurrently, it would not be able to merge. The documentation was confusing, and we failed to set it up properly a few times. We also had trouble with incompatible versions between Unity and the Google VR SDK. The SDK we originally had was out of date. Additionally, another challenge we faced was getting the driver into the car and being able to make him move in a realistic manner.

9.3 Open Issues & Ideas for Solutions

N/A

10. System Manuals

10.1 Instructions for System Development

In order to develop the application, the environment must be set up. After the required steps are completed, the project must be opened in Unity. From there, any part of the system can modified.

10.1.1 How to Set Up Development Environment

In order to develop the application, the developer must have Unity 5.4.1f1 installed as well as Git in order to pull from the repository. Once pulled, opening the project in Unity will allow for additional development.

10.1.2 Notes on System Further Extensions

10.2 Instructions for System Deployment

Steps to build and export to Android:

1. Select File > Build Settings
2. Select the platform as Android, then switch platform
3. Select Player Settings, in the resolution and presentation tab, select landscape left and use the 32-bit display buffer
4. Select Other Settings, change minimum API level to be Android 4.4 KitKat (API level 19)
5. Select Build to create APK

10.2.1 Platform Requirements

In order to build and deploy the application, Unity is required. Along with that, the Android SDK and Java SDK must be installed as well.

10.2.2 System Installation

To install on Android, the APK must be downloaded. After downloading, it can be installed and then started.

10.3 Instructions for System End Users

N/A

11. Conclusion

11.1 Achievement

11.2 Lessons Learned

11.3 Acknowledgment

12. References

[1] MSDN, C# Programmer's Reference, Accessed on 10/21/2016
https://msdn.microsoft.com/en-us/library/618ayhy6(v=vs.71).aspx

[2] Unity, Unity Scripting Reference, Accessed on 10/21/2016
https://docs.unity3d.com/ScriptReference/

[3] Unity, Unity Manual, Accessed on 10/21/2016
https://docs.unity3d.com/Manual/index.html

[4] Unity, Unity Test Tools, Accessed on 10/21/2016
https://unity3d.com/learn/tutorials/topics/production/unity-test-tools

[5] Unity, Unity Community, Accessed on 10/21/2016
https://forum.unity3d.com/

[6] Google, Google VR SDK for Unity, Accessed on 10/21/2016
https://developers.google.com/vr/unity/