Efstathios Efstathiou - w15029887

Computer Sciences with Animations, Graphics and Vision

KV6002: Team Project and Professionalism

Supervisor: Edmond Ho

2nd marker: Lynne Connis

Contents

[1.0 INTRODUCTION 3](#_Toc6930506)

[2.0 SUBSYSTEMS 3](#_Toc6930507)

[3.0 TESTING STRATEGY 5](#_Toc6930508)

[4.0 HARDWARE 7](#_Toc6930509)

[5.0 ENVIRONMENT REQUIREMENTS 7](#_Toc6930510)

[6.0 RISKS & ASSUMPTIONS 7](#_Toc6930511)

[7.0 REFERENCES 8](#_Toc6930512)

## SOFTWARE TESTING REPORT

## 1.0 INTRODUCTION

This document analyses the overall approach and the different components on the software testing of Oculus Rift product. As a team, we must produce software capable of satisfying the requirements of our customer and creating a 3D crime scene environment with possibilities to be evolved and work in professional forensics labs. The testing strategy is an essential part of this document and so the results, in the following section there are the different components that support the pre-testing, during testing and post-testing procedure and development.

## 2.0 SUBSYSTEMS

* 3D Scene – Environment by Luke Rose
* User Interaction by Hassan Mohammad
* User Interface by Zoe Irwin

2.1General approach:

The 3D scene is a section that the developer can work on full independence without depending on another subsystem, it should be the first thing that will be created so the others subsystem can have an environment to work. Although it is not able to finish as the user interaction must work correctly with the scene, and there should be a critical collaboration between the two subsystems.

User Interaction have a meaningful role as our product is based on the representation of a crime scene and the user must be able to interact on the scene. A scene is requiring to be there to start working on the specific subsystem. At this stage, it should be a high level of professionalism as it is something that depends a lot on another subsystem.

User Interface looks the most independence section, but it is the primary connection between everything of the final product, the structure of it is based on the interface, an error on the interface can cause dramatically bad software errors or endless loop while the user has no power to change it.

2.2 Subsystem features & objectives to be tested.

1. 3D Scene – Environment

Functional requirements:

* Utilize 3D assets.
* Create story driven objects within the scene.
* Scale the room to the field of view of the Oculus Rift.
* Use high-resolution textures.
* Create a fully enclosed room.
* Create objects that the user can look under and around.
* Include light sources in specific areas.

Non-functional requirements:

* Re-create the living room for Manor House in Coach Lane Campus.
* Create a realistic feel for the environment.
* Elicit an immersive feel.

1. User Interaction

Functional requirements:

* Integrate the Oculus Rift hardware controls with the VR application for the user’s control.
* Allow 3D objects to be manipulated within the VR application by allowing tasks such as rotation and a close-up inspection.
* Produce a control scheme/mechanism for the user to navigate and move through the 3D environment while wearing and using the hardware required.

Non-functional requirements:

* Improve immersion by using real-life interactions as the basis model for how the user should be expected to interact with the 3D environment and the objects within it.
* The application should include feedback mechanisms to notify when the user has interacted with an object.

1. User Interface

Functional requirements:

* Allow basic navigation through the main menu screen.
* Display and instruct the user on how to navigate around the scenes.
* Make user interfaces more interactive with the use of sound & graphics.
* Sub-menu interfaces for the user to interact within the scene.

Non-functional requirements:

* Make user interfaces as user-friendly.
* Make user face consistent.
* Make user face efficient, quick response times.
* Make a high-quality interface.

All the above subsystems have their specific requirements that are already listed on group documentation of Term of Reference.

## 3.0 TESTING STRATEGY

3.1 Scope

Considering the objectives of the product and the results that we are looking for as a team, the testing strategy aims to cover the process professionally, especially when we have a customer expecting to receive a product that fulfils the requirements. Our working environment and conditions are not limited to the student area as we should provide a system fulfilling all our objectives and satisfy a real customer, the Head of Forensics department of Northumbria University.

3.2 System and Integration Testing

Definition:

Integration testing is the phase of the individual parts are combined to create the very first version of a complete system and test it under those circumstances where unexpected errors may occur or poor connection among the subsystems.

Methodology:

System and Integration testing methodology are to approach the system as a combined integrated software that the differences among subsystems are not easy to spot. A smooth and flawless product is what we are looking for and testing it is based more on perception and not if there is complete the objectives.

3.3 Performance and Stress Testing

Definition:

Evaluation of quality of the product defines the performance testing to ensure that the system performs as expected based on objectives and from a general point of view as developers and users. Stress testing on performance happens under adverse conditions following a different usage of the system to detect errors or any other unwanted results while the system is running. It is not what the typical user will do but is necessary from the creator to fill any gaps and avoid dissatisfactions. Stress testing is commonly before beta versions, so the team will manage any problems that could occur.

Methodology:

Performance and stress testing generally define the foundations of a system and objectives requirements. Method on performance is more like an everyday job as it should always be our priority, no matter what we are working on it should be acceptable from any point of view at least to satisfy the customer on an early version. Stress testing is method follow a different approach as we must think outside of the box and push the product to their limits of its environment and in general on the platform that we are using as a computer. It is essential to know the capabilities of our system and the weaknesses even if the user will never notice targeting to minimise errors as much as possible under stressful conditions.

3.4 User Acceptance Testing

Definition:

The purpose of user acceptance testing is to confirm that the system is ready for operational use, it is pre-beta testing, and it can execute on individual subsystems as well. It is a combination of performance, stress and overall system testing which always following the objectives requirements and the expectations of the customer.

Methodology:

The method for user acceptance testing is basically to put ourselves out of developer’s role and look at the product as a customer, of course, performance and consistency must be on point but furthermore knowing the objectives we must create a system that satisfies user’s demands.

3.5 Regression Testing

Definition:

Regression testing is the selective retesting of a system or a specific component of a subsystem to verify that the modifications have not caused any other effects and that system or component still works as specified in the requirements. It will only be applied in case of dysfunctionality or a similar problem.

Methodology:

To simplify the regression testing process, we will always compare the previous version before the modification and then right after any change. If a problem arises during a modification, the team member that is responsible will probably separate the actions in order to identify the source of the error.

3.6 Beta Version Testing

Beta version testing is the usage of the system by the creators or a relevant team before the final submission to the customer, is also known as the pre-release testing and it comes after the user acceptance testing. Ideally, the team or the tester on this stage need days to interact and gets into the customer's eyes, by the end of the process the results must answer the question if the product is ready to be delivered. In our case, unfortunately, beta testing will not proceed as it should be, the main reason is the environmental testing requirements are higher than usual. It will be executed to the maximum level but also the time we had to work with our hardware in a specific environment was limited and shorter than we expected as a team. Three members of our team are using the same equipment to build the product, and it seems hard to achieve proper beta testing.

4.0 HARDWARE

Oculus Rift is a virtual reality headset with surround audio headphones attached with, two camera sensors called Constellation are responsible for user’s positional tracking and a pair of wireless motion controllers to interact on the virtual environment are the rest of the equipment.

## 5.0 ENVIRONMENT REQUIREMENTS

The main software creation is taking place under environmental circumstances. The developing and the testing process are done in the same period as the hardware we are using is demanding especially about the environment. It is good to mention that as a group of students we daily loan the Oculus Rift from university facilities as it is too expensive to get one for ourselves. Firstly, the testing process should happen in a specific room at university because our hardware entails with a powerful computer to work correctly without any delays or inadequate graphical response. Some of the system features can test as they are working as standalone without any addition of others work like the User interface subsystem, at least in most parts. Some others are highly depending on other subsystems, for example, User interaction cannot be tested if there is no environment at least in most parts.

Communication has a critical role at this point of the developing and testing because of the high restrictions, and it is one of the main reasons why testing subsystem support the excellent communication as the first objective of testing. Another critical point is the physical space that we need to execute the testing and the developing of the project, the virtual reality headsets need a clean place of the room to work, and as we are working at university in a regular room which makes it harder. Although Oculus Rift seems to respond very well even in limited space, the Constellation setup is slow, and sometimes we must change computer or readjust the sensors a couple of times, is time-consuming and happens every single time.

## 6.0 RISKS & ASSUMPTIONS

Two of the highest risks are the synchronisation among the group members and the deadlines on deliverables that we already agreed as a group. Both can cause significant problems on the functionality, and the progress of the team especially the testing subsystem will face a massive problem as the work that is supposed to be done depends clearly on others and mainly based on deadlines. Separately synchronisation can be described as an excellent plan for a brilliant groupwork, and there are no such errors on testing even if the team harmonisation is missing, but the deadline deliverables are there. On the other hand, if the group is not following the schedule and even one member is far behind, then, the testing will be incorrect on the specific subsystem and weak on other related subsystems. It is clear enough that it may cause chaos if a team fail to follow the deadline schedule and deliverables.

## 7.0 REFERENCES

[1] Spillner, A., Linz, T. and Schaefer, H., 2014. *Software testing foundations: a study guide for the certified tester exam*. Rocky Nook, Inc.

[2] Hass, A.M., 2014. *Guide to advanced software testing*. Artech House.