Architectural Design

Matthias Tavasszy
4368401

Arjan van Schendel
4366212

Luke Prananta
4288386

Jasper van Esveld
4372581

Bart Ziengs
4391799

June 17, 2016

Contents

Chapter 1

Introduction

1.1 Design Goals

The design goals are things we want to accomplish in terms of our design of the product. These can be divided in several categories.

Fully tested and integrated

Every week a working build is produced with above 80% test coverage and integration with Unity Cloud Build. We must ensure that a user will not experience any system breakdowns or bugs. The people who will use our system are often mentally fragile and any negative experience can have long lasting or permanent negative consequences.

Availability

The product we aim to deliver must be available at preferably any time. For that reason we must pursuit an solid implementation that is easily modifiable if any additional enhancements are demanded.

Performance

Immersion is a key part of virtual reality and to get the best experience we need to deliver a product that can maintain a consistent and high frame rate. Since the Oculus Rift can display images at a refresh rate of 90Hz our product needs to be able to deliver a minimum frame rate of 90 fps.

Precision

We need to allow users to pick up items within the virtual world. Manus VR and the Leap Motion will allow these interactions with high precision tracking of the hand and fingers. The Kinect will be used to track the rest of the body for extra immersion.

Reliability

Immersion can be compromised when we lose tracking of a part of the body. We can't prevent these moments when tracking is lost but since we have multiple tracking devices we need to use data of another device to fill in the gaps. If no data is available for the current position we can still predict the position using using old tracking data.

Chapter 2

Software architecture views

This chapter lists both the functional and non-functional requirements The MoSCoW method is a prioritization technique, we divided our requirements into must, should, could, and won't haves.

2.1 Programming languages and programs

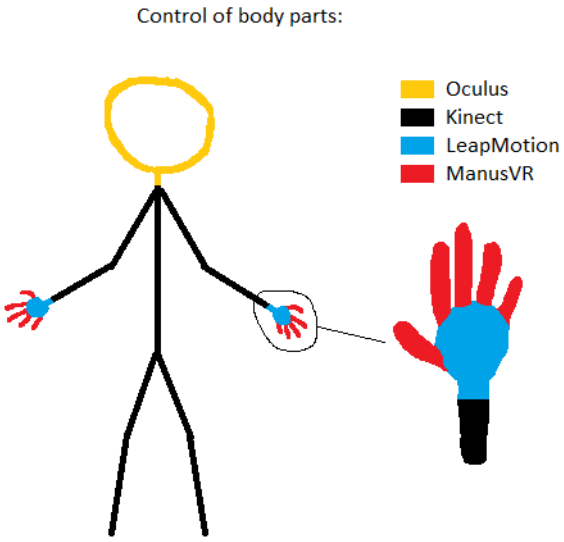
The program will run on Unity3D, version 5.3.4p1. The program will be developed in the C# language using Microsoft Visual Studio as the script editor.

2.2 Continuous Integration

To test and build the Unity program, we make use of a service called Unity Cloud Build provided by Unity Technologies.

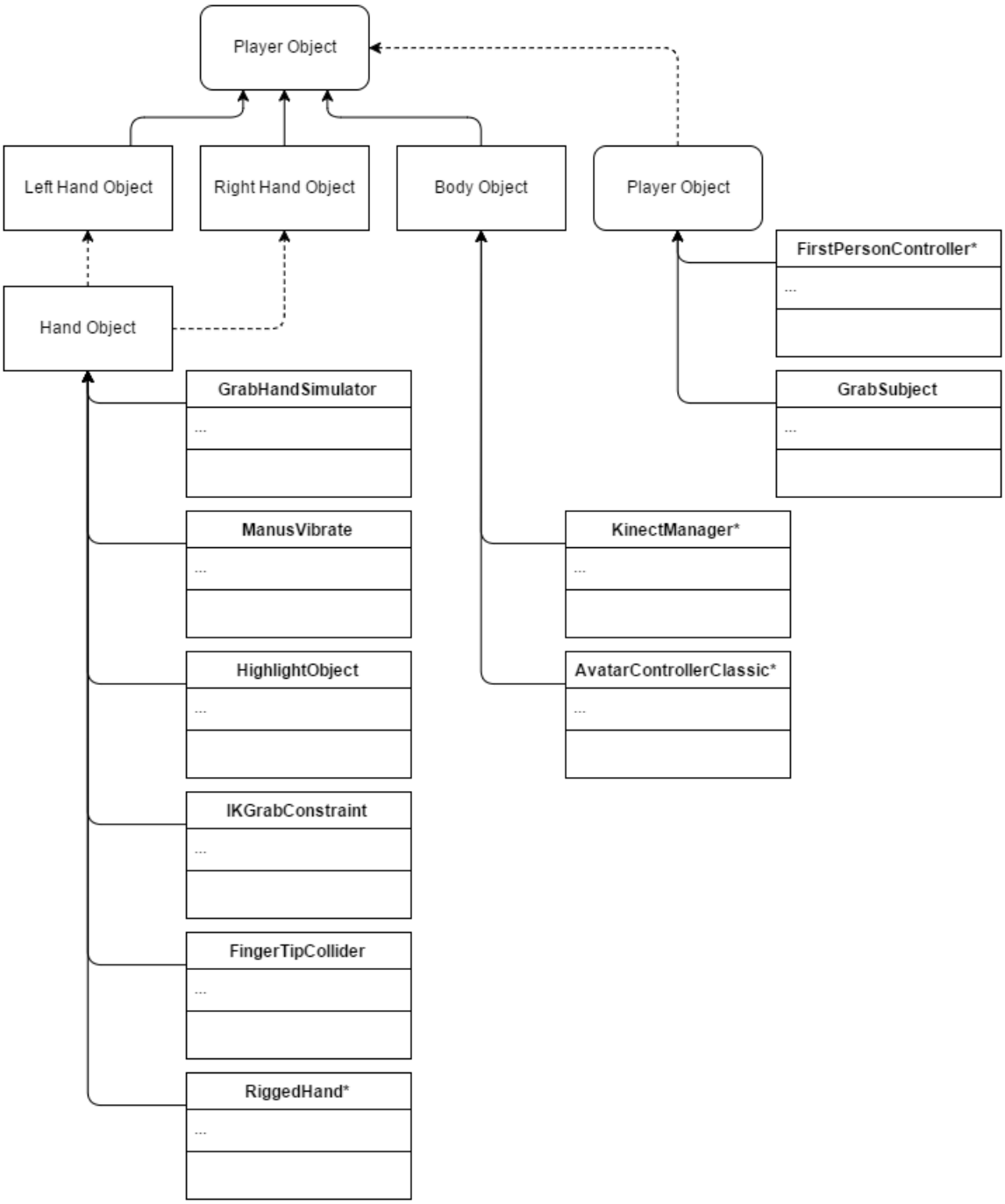
2.3 Hardware/software mapping

The system uses several hardware components. A computer will run the program through Unity3D, while the Kinect, Leap Motion and Manus VR will provide measured tracking data for the program while active. The hardware used for tracking will be connected to the computer.

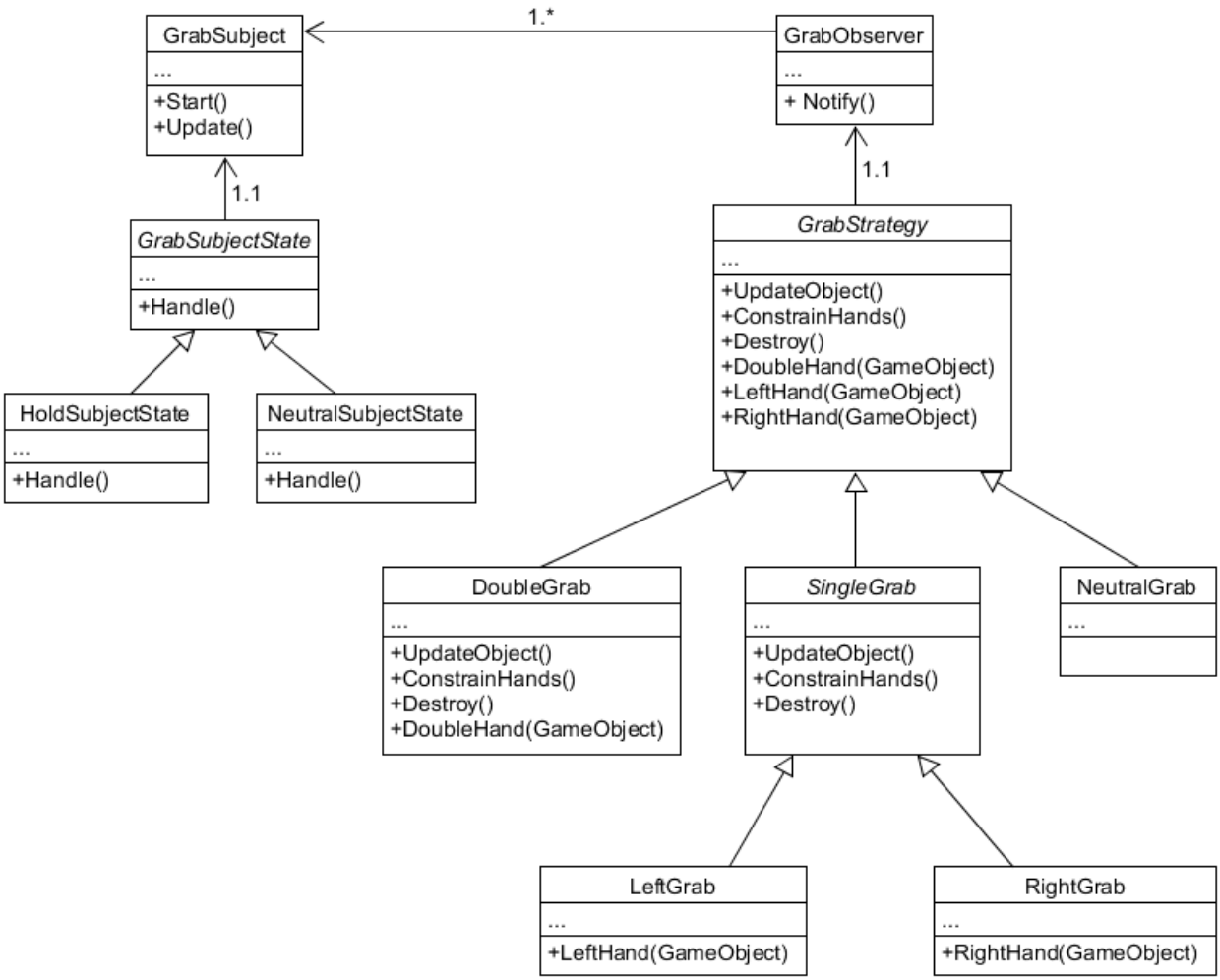


This figure shows how each piece of hardware controls each part of the virtual body.

2.4 General System Overview



Classes marked with * are third-party.



Our grab behavior implements several design patterns. We make use of a observer pattern, strategy pattern and state pattern.

2.5 Version control

Version control will be provided by git. To assure good code quality, pull requests will be used to review changes before merging these changes to the main branch.

2.6 Concurrency

To ensure concurrency between the several hardware and software components we make use of, we only use them in a setting as suggested by the product owners and use strictly the versions of each program that is known to support concurrent use of software.

2.7 Persistent data management

There is a very limited amount of data to be stored within this project. The only thing that stays persistent is the body model, and the Unity map to some extend.

2.8 Tools

The most important tools that were used for this project are explained below.

StyleCop

StyleCop is the visual studio equivalent of CheckStyle. Just like CheckStyle it enforces the standards within Csharp for example classes and functions must start with a capital letter. Besides that StyleCop also ensures that comments meet certain requirements like documentation on return values and parameters. We used StyleCop to improve readability during the development process, this not only helped us during the development but will also help CleVR when we deliver the final product.

CodeMaid

CodeMaid is a very handy tool that helps solve StyleCop errors (white space, method order etc.). CodeMaid was recommended by CleVR but we ended up not using it all that much because most of the problems that CodeMaid helped to fix were easy to do by hand.

GhostDoc

To speed up the creation of documentation we used GhostDoc which generates the basic structure for a method or class with a simple shortcut.

CloudBuild

CloudBuild is the Continuous Integration solution for Unity. It works with any Unity project can be connecting with a git repository. For each change within the project it will than automatically build and run all tests resulting in succeeding and failing build. There is however one downside and that is the time for each build when using the free plan. The average build time for our project was more than one hour which meant that using CLoudBuild for things like pull requests became basically impossible.

Other used tools:

- VSSpellChecker
- SandCastle
- ProPowerTools

2.9 Organizational Tools

- Google Drive
- Slack
- Trello

Chapter 3

Glossary

Unity3D

Unity3D is a cross platform game engine developed by Unity Technologies and used to develop video games for PC, consoles, mobile devices and websites. In this context the engine will be used to develop the program and environment.

Manus VR

The Manus glove is the first consumer glove specifically designed for virtual reality. It tracks hand movement using a combination of high tech sensors all contained inside the glove. With this technology, the Manus glove provides accurate and reliable hand and finger tracking.

Kinect

Kinect is a webcam like motion sensing input device by Microsoft. It uses two cameras to track the user, as well as an infrared sensor to measure the distance from the user, and to create depth. It can be used for tracking rough movements of the human body, but fails to track small, precise movements, like moving separate fingers.

Leap Motion

Leap Motion is a device that can be used to track the user's hands and fingers using infrared LEDs and two cameras. The main difference between the Leap Motion and the Kinect is the smaller range of the Leap which in return results in increased precision. Now with the increasing interest in virtual reality the developers of Leap Motion implemented a virtual reality mode where the sensor is put on the virtual reality headset (the Oculus Rift in our case).

Oculus Rift

VR System developed and released by the Oculus Company. This system includes a Head Mounted Display. It has a gyroscope for accurate rotational tracking and limited positional tracking capabilities.