Alpha

Communication in Software, an Introduction

Raja Williams

December 2023

Contents

Chapter 1

Introduction

A core part of engineering, especially modern engineering, is the communication of ideas to others, whether it be for designers to manufacturers, or even designers to designers. Software, like hardware, is to be designed and engineered, and most of all, manufactured as in writing it. The purpose of this project is to demonstrate the importance of communication in modern software engineering, and create an application second.

The primary inspiration behind Alpha was the Apollo Space Program initiated under the Kennedy Administration. Not only did the NASA Manned Spacecraft Center team have time pressure, the nature of the program required an intense testing protocol, to ensure the safety and security of the astronauts. This required a high level of communication, requiring contractors and subcontractors to test to the specific guidelines laid out by the many engineers and boards assembled at NASA.

This pressure found in the Apollo program is reflected in how Alpha's design and mission is guided. Before any of the software was designed, principles guiding design were created. As well, each component, when fully redesigned or otherwise changed in an interface-breaking way, will be designated as a new "model" or "mark" of the component. As an example, the first version of the Draw component is called "Draw: Mark I."

1.1 Principles

To achieve a high quality final product, principles must be laid out to guide the creation of both hardware software. These principles borrow a lot from the hardware design principles during the Apollo space program, as written in NASA SP287.

1. Build off of established software.
2. Minimize interfaces between components.
3. Make each component reusable.

4. Small steps go a long way.
5. Use as much experience as possible from each step.
6. Make each component reliable.

Using established software ensures reliability and, in some cases, performance of a component. This allows for an increased focus on the more difficult and less tedious portions of software design, which is integral for a team on a time crunch or a team without sizable manpower or wherewithal. Established software is also easy to come by in the form of free, open-source software.

Minimization of interfaces and making components reusable go hand in hand, especially in software. By minimizing the interfaces between components, it is inherently easier to pull a component out of it's codebase and use it inside of another codebase. Another part of this is ensuring that each component has isolated build-system files. As in each build-system is separated and can build freestanding without needing to be inside the Alpha codebase.

1.2 Mission Statement

The purpose of the Alpha program is to simulate and display a possible rocket launch from Earth to Mars. The required software to achieve this are as follows:

1. A n-body simulation of the solar system.
2. Method to calculate the trajectory required to reach Mars by rocket.
3. Simulation of rocket thrust and the subsequent decrease of mass.
4. Method to open and manage the window and drawing context.
5. Renderer of the n-body simulation.

For Item 1, to maximize the re-usability of this component in future projects, this piece will be it's own component, codenamed n-body. In the source code of Alpha, this component can be found under the `nbody` folders or repositories.

For Items 2 and 3, as these are closely related, these two pieces will be combined into their own component, codenamed Rocket. In the source code of Alpha, this component can be found under the `rocket` folders or repositories.

For Item 4, to maximize the re-usability of this component in future projects, will be it's own component, codenamed Draw. In the source code of Alpha, this component can be found under the `draw` folders or repositories.

For the final Item 5, this piece is closely related to however not fully required for the n-body simulation. As a result, it will need to be a separate component to ensure that a dependency on Draw for n-body is not required. This component will be called the n-body Renderer. In the source code of Alpha, this component can be found under the `nbodyrenderer` folders or repositories.

A majority of these components require a substantial amount of linear algebra. To build off of established software and maximize reliability of all components, glm will be used as a dependency for all of the components listed.

Chapter 2

Draw: Mark I

To facilitate the management of the window and drawing context, Draw: Mark I was the first iteration of the Draw component.

2.1 Purpose

The Draw module has to achieve and manage several different tasks.

1. Window creation and deletion.
2. OpenGL context creation and deletion.
3. Rendering objects.
4. Main window interaction loop.

To achieve window creation and deletion, GLFW is used. This ensures reliability as GLFW is a well-tested and established library for window and context creation. To manage the context, a generated version of glad is used.

2.2 Usage

2.3 Tests

To ensure that rendering is fully reliable and in working order, a multitude of tests were created. These can be seen in Figure 2.1.

	Dependencies	Purpose
`window00`	`drawmki`	To demonstrate window creation and main window loop.

Figure 2.1: Draw: Mark I tests and purposes