Oregon State University
College of Engineering

CS Capstone Requirements Document

October 26, 2019

OSU NASA USLI Competition

Prepared for

NASA, OSU AIAA

Dr. Nancy Squires

Prepared by

Group CS78

Team Rocket

Manuel Ochoa-Botello

Milo Chase

Nolan Nichols

Abstract

This document outlines the client requirements for the Oregon State University (OSU) Capstone Project: "OSU NASA USLI Competition." This document is intended to provide further insight and describe what the final product should do and how it will be constructed.

CONTENTS

1 Definition and Description 2

1.1 Purpose . . . 2
1.2 Scope . . . 2
1.3 Definitions . . . 2
1.4 References . . . 2
1.5 Overview . . . 2

2 Overall Description 2

2.1 Product Perspective . . . 2
2.2 Product Functions . . . 2
2.3 User Characteristics . . . 3
2.4 Constraints . . . 3
2.5 Assumptions and Dependencies . . . 3
2.6 Apportioning of Requirements . . . 3

3 Specific Requirements 3

3.1 Functions . . . 3
3.1.1 Rover Software . . . 3
3.1.2 Rocket Avionics . . . 3
3.1.3 The Blade Extending Apogee Variance System (BEAVS) . . . 4
3.2 Performance Requirements . . . 4
3.3 Design Constraints . . . 4
3.4 Software System Attributes . . . 4
3.4.1 Reliability . . . 4
3.4.2 Availability . . . 4
3.4.3 Security . . . 4
3.4.4 Maintainability . . . 5
3.4.5 Portability . . . 5

4 Verification 5

4.1 Functional Requirements . . . 5
4.2 Performance Requirements . . . 5
4.3 Design Constraints . . . 5
4.4 Software System Attributes . . . 5

5 Appendices 5

5.1 Assumptions and Dependencies . . . 5
5.2 Acronyms and Abbreviations . . . 5

6 Gnatt Chart 6

1 Definition and Description

1.1 Purpose

This document is intended to provide a narrow scope of the flight telemetry system which will be implemented within the NASA United States Launch Initiative (USLI) rocket. The intended audience for this document is for the USLI team, system developers, and NASA Ssponsors.

1.2 Scope

The core product being delivered is a high-altitude rocket and rover that will complete a series of tasks and missions as defined by Nasa. The flight telemetry sub-team is responsible for developing code that will interface with hardware located within the rocket. it will be responsible for guiding the rocket to a predetermined altitude.

1.3 Definitions

- Hardware: The machines, wiring, and other physical components of a computer or other electronic system
- Software: The programs and other operating information used by a computer.
- Telemetry: The process of recording and transmitting the readings of an instrument

1.4 References

1.5 Overview

The following chapter provide further insight on the requirements and details of the NASA USLI rocket.

2 Overall Description

2.1 Product Perspective

The OSU NASA USLI rocket and rover are artifacts intended to meet the needs and requirements outlined by NASA. The rocket fuselage will house the majority of the system interfaces including the rover; it is responsible for protecting the assets within the fuselage and withstand any challenges that come forth during the NASA USLI competition. Hardware within the artifacts will be used to assist and interface with other components and provide a way to interact with the outside world. Such hardware is responsible for capturing data, moving or altering internal and external components,and allows software to control the hardware systems and components on board. The software created for this project is intended to be used for several systems- the on board flight telemetry software will be responsible for intercepting and analyzing incoming hardware controller data to support the rocket while in flight. The payload and recovery software system is responsible for interacting with certain hardware to successfully deploy the rocket rover and safely land the rocket.

2.2 Product Functions

The rocket and rover will used to simulate and demonstrate the ability to navigate a mission to space within the NASA USLI competition. The rocket will deploy at ground-level, reach a predetermined altitude set by the OSU USLI team,launch the rocket recovery system, and deploy the rover. The rocket and rover will land on the ground; the rover is then required to collect a scientific sample within an undisclosed area, and move outside of the sample area to complete the mission

2.3 User Characteristics

The target group of the software product is the OSU NASA USLI team. Their intent is to utilize the software to assist the rocket and rover in navigating through the challenges during the NASA USLI competition. This target audience will benefit the most from the product as it will allow them to overcome certain limitations that hardware alone cannot provide. The second target audience is NASA- The NASA organization sometimes uses student designs for further research within the aerospace industry

2.4 Constraints

The software will be subject to several constraints throughout development:

- The software will abide to memory constraints of physical hardware
- The software must be able to handle fast operations as it will be working with real-time data
- It must handle error correction quickly
- It must relay data over a network

2.5 Assumptions and Dependencies

One assumption is that the software will be located within consumer grade development micro controllers, interfaced with various hardware components. Memory limitations are currently unknown, however, it is assumed that the software will be developed with the best time efficiency in mind. Many consumer grade micro controllers will supply 4Mb of working memory, software designs will take this into consideration. The on board software will be written in the most efficient language that is compatible with the micro controllers provided to the team.

2.6 Apportioning of Requirements

Development of the software systems required for the rocket and rover will be software development for the rover, the rocket avionics and the Blade Extending Apogee Variance System or BEAVS.

3 Specific Requirements

3.1 Functions

3.1.1 Rover Software

The UI for the rover software will be done in C#. The UI must display the compass correctly and have the compass point into the correct direction. The UI is also to display the coordinates of the drone and have the control to lower and raise the auger to collect the sample. Team Rocket after receiving the rover from the Electrical Engineers will make sure all the sensors involved are able to communicate with one another.

3.1.2 Rocket Avionics

Team Rocket will look into getting the altimeter, GPS and other avionics to all talk to each other and to the user to be able to accurately record the altitude of the rocket and correctly eject the parachutes of the rocket. The system will send information back to the user and be connected to the rover software in the sense that the rover software will display the location of the rocket and payload.

3.1.3 The Blade Extending Apogee Variance System (BEAVS)

The Blade Extending Apogee Variance System or BEAVS is a new system that the team is looking to implement this year which will have a system that calculates the speed at which the rocket is flying. Depending on the speed of the rocket, the system will either extend or retract the fins in the back to be able to hone in on the proposed altitude of 4000ft. BEAVS will be flashed with a program that will be taking in data from the altimeter, barometer, micro motor, and encoder that will pass in data related to the rockets velocity and altitude and depending on the results that is returned by the program, the system will determine whether or not to extend the fins in the back. The language that this program will be coded in is Arduino to flash the board and Python to be able to do simulations and scripting to test and see the values that would be returned. The expected finish date for this system is early January 2020.

3.2 Performance Requirements

The system that Team Rocket is developing should be able to be easy to use and work 99.999% of the time. Reasoning for high expectations for the system is that if one of the systems is to go offline, it could cause a huge problem with the whole launch in which it will not help achieve the proposed altitude. To be able to do this, constant testing will be done for each piece to ensure that for every trial of the rocket launch, the system is to succeed and Team Rocket will look for any edge cases that could cause failures in the future.

3.3 Design Constraints

Constraints for the program are languages that work well when programming for system boards. Research is done to ensure the languages that will be used to develop different part of the system. For the GUI of the rover, C# will be used. To flash boards, Arduino will be used since we are working with teensy boards.

3.4 Software System Attributes

3.4.1 Reliability

The software that will be made for each piece of the rocket needs to be able to accurately perform to be able to help succeed the mission by NASA USLI. For the rover software, the software must accurately display the correct video from the rover and be able to correctly control the rover without any bugs. For rocket avionics, all the different components of the rockets are to communicate with each other. If one of these systems fail, it could very well cause major issues in the rockets and other components. Lastly, BEAVS has to be able to accurately take in the correct value to be able to determine if we need to increase or decrease the drag of the rocket.

3.4.2 Availability

Since the simulator of the rocket is coded in Python, the only thing that will be needed is a computer that is able to execute Python code. Since we are going to use teensy board, the boards that we are going to be using are teensy 3.6 boards, therefore this project must work on equivalent boards.

3.4.3 Security

No sensitive information is going to be passed into the program so no focus on security is needed.

3.4.4 Maintainability

Since NASA ULSI, the programs that will be coded for this project must be well documented to be able to pick up by next year's team to continue development and improve the system year after year.

3.4.5 Portability

With this project built with Python and Arduino, it helps porting the code used for this project and pass to other projects within OSU AIAA program.

4 VERIFICATION

4.1 Functional Requirements

To ensure each functions work, we will constantly test the rover software and make sure the camera of the rover is working 100%. We will constantly use the controls to be able to make sure the rover is able to move well in any direction we control the rover. To test the rocket avionics we will constantly be using equations for the rocket to be able to make sure the values passed by the components are correct. To be able to test BEAVS we will constantly test equations gathered for this system and see if the correct values that we get out of the equation is the correct result we get when we test the rocket.

4.2 Performance Requirements

The performance will be tested by checking and see if we get the correct results each time we have a test launch.

4.3 Design Constraints

Make sure the language used for the board and simulating is done in Python and Arduino. As well the GUI being done in C#.

4.4 Software System Attributes

Verifying system attributes will all done through user testing.

5 APPENDICES

5.1 Assumptions and Dependencies

To make the software usable, the assumption Team Rocket will make is the user of the rockets is familiar with the equations used for NASA USLI and are familiar with teensy boards.

5.2 Acronyms and Abbreviations

- GUI: Graphical User Interface
- BEAVS: The Blade Extending Apogee Variance System
- OSU: Oregon State University
- NASA: National Aeronautics and Space Administration
- USLI: United States Launch Initiative
- AIAA: American Institute of Aeronautics and Astronautics
- UI: User Interface

6 GNATT CHART

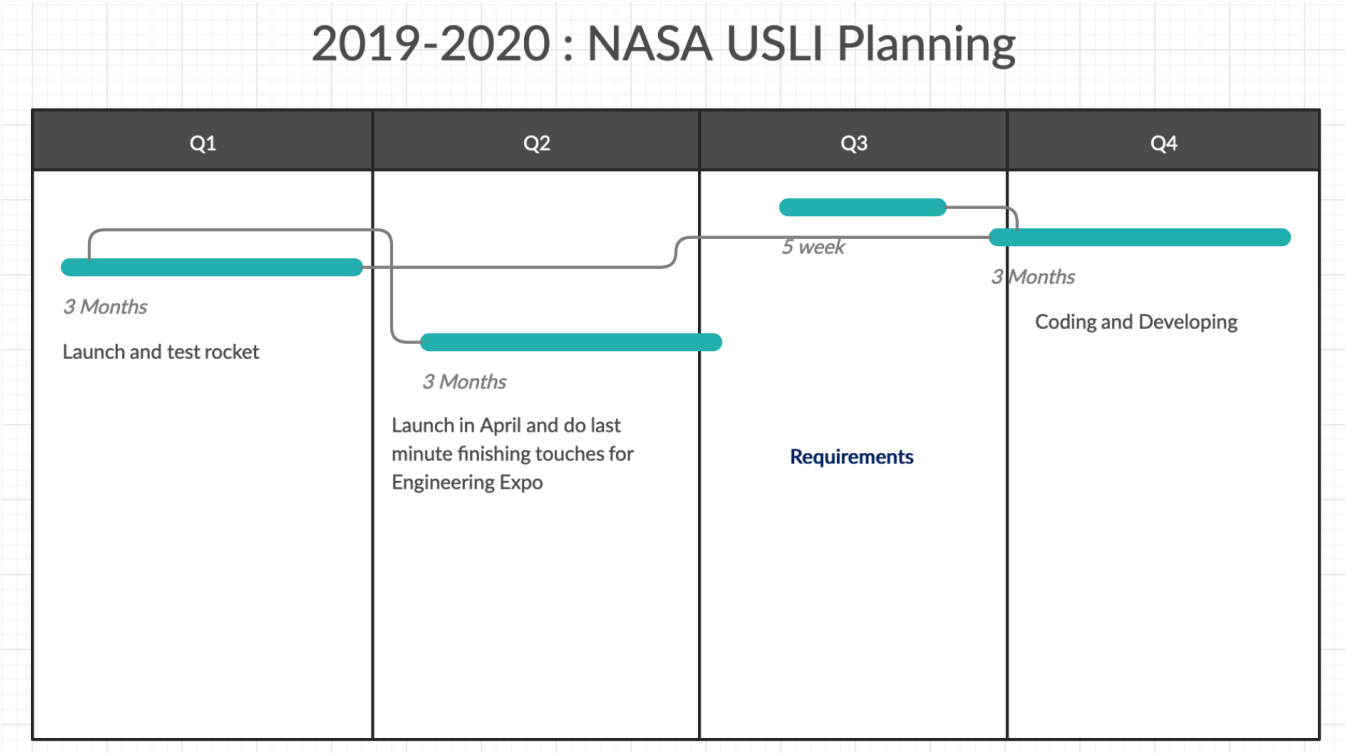

Fig. 1: Gnatt Chart