SOFTWARE REQUIREMENTS SPECIFICATION

for

The ARLISS Project

Version 1.0

Steven Silvers
Paul Minner
Zhaolong Wu
Zachary DeVita

Oregon State University
October 26, 2016

Contents

Revision History

Name	Date	Reason For Changes	Version
21	22	23	24
31	32	33	34

1 Introduction

1.1 Purpose

The purpose of this document is to provide a thorough and verbose description of our teams software which we will be developing for use in the ARLISS International Competition; the acronym ARLISS referring to 'A Rocket Launch for International Student Satellites'. The document will explain the purpose and features of the system, as well as, objective of the system, and the constraints under which it must operate. This document is intended for any stakeholders in the project, i.e. clients, instructors, and engineering collaborators, as well as, for the software developers who will be creating the system.

1.2 Document Conventions

This document follows IEEE Format. Bold-faced text has been used to emphasize section and sub-section headings. Highlighting is to point out words in the glossary and italicized text is used to label and recognize diagrams. Requirements will be prioritized into two categories; High-level requirements will be denoted with a heading of 'Obligatory Requirements', while stretch-goals will be denoted with a heading of 'Secondary Requirements'.

1.3 Intended Audience and Reading Suggestions

This document is to be read by the software development team, the extended team of engineers who will be collaborating on this project, course instructors, and the client for the project. The document will be publicly accessible as well, and may be read by anyone viewing our exhibit. The document can, and will, be read by our peers in software development, as well as, engineers of all disciplines.

Overall Description – The general public, as well as, the media may find the overall description to be most pertinent, and to give an adequate and effective overview of the ARLISS Project.

System Features – Developers, as well as, the extended team of engineers who are collaborating on this project will require a comprehensive understanding of the system features in order to integrate their design, and, additionally, for testing purposes.

Nonfunctional/Functional Requirements – The computer scientists developing the software, and the electrical and mechanical engineers developing the hardware will need to utilize, and adhere to, the requirements.

1.4 Project Scope

Our team of computer scientists will be developing software to operate an autonomously navigated satellite to a specified set of coordinates during the international competition. Our team's satellite will be ejected from a rocket at approximately 12,000' AGL, and fall safely to the surface of Earth with the aid of a parachute. During the estimated 15 minute "hang time" that the rocket spends in the air, there will be barometric reading taken every 1000ft by the satellite. Once the satellite reaches the ground it must detach from the parachute and circumvent any potential restraint created by the canopy or suspension lines of the parachute. Once moving, the autonomous satellite must utilize its system of cameras and radar to detect obstacles so that it may navigate around them. GPS will be necessary for the satellite to find the coordinates.

The most imperative objective of the software is to safely guide the satellite to its intended destination. This is a requisite, but the secondary objective of taking barometric readings is an additional goal which our group is confident in completing.

1.5 References

2 Overall Description

2.1 Product Perspective

2.2 Product Functions

2.3 User Classes and Characteristics

2.4 Operating Environment

2.5 Design and Implementation Constraints

2.6 User Documentation

2.7 Assumptions and Dependencies

3 External Interface Requirements

3.1 User Interfaces

3.2 Hardware Interfaces

3.3 Software Interfaces

3.4 Communications Interfaces

4 System Features

4.1 Parachute Deployment

4.1.1 Description and Priority

This system will need to detect when the satellite has been deployed from the rocket and activate the parachute module to carry the CanSat safely to the ground.

4.1.2 Stimulus/Response Sequences

4.1.3 Functional Requirements

4.2 Change to Drive Mode

4.2.1 Description and Priority

This system will need to get the rover ready to drive once it has landed on the ground. This could involve activating various servos within the CanSat to expand the rover from the payload container.

4.2.2 Stimulus/Response Sequences

4.2.3 Functional Requirements

4.3 Sensor and Motor Array Initialization

4.3.1 Description and Priority

The driving sensors and motors made need first time initialization before it can begin driving. This would include locking on to the target GPS coordinates, and preparing any sensors needed for the obstacle avoidance system.

4.3.2 Stimulus/Response Sequences

4.3.3 Functional Requirements

4.4 Obstacle Avoidance System

4.4.1 Description and Priority

Once the rover is locked on to the target GPS location, it will need to autonomously drive to the target location. This could be anywhere from less than a kilometer to fourteen

kilometers depending on how the parachute is carried. There will be obstacles between where the rover lands and the target location such as rocks that will need to be avoided by our rover. This system will take input from the rover's on-board sensors and use that data to control motor function so that the obstacles will be avoided.

4.4.2 Stimulus/Response Sequences

4.4.3 Functional Requirements

4.5 System Feature 5 (and so on)

4.5.1 Description and Priority

4.5.2 Stimulus/Response Sequences

4.5.3 Functional Requirements

5 Other Nonfunctional Requirements

5.1 Performance Requirements

5.2 Safety Requirements

5.3 Security Requirements

5.4 Software Quality Attributes

5.5 Business Rules

6 Other Requirements

6.1 Appendix A: Glossary

6.2 Appendix B: Analysis Models

6.3 Appendix C: To Be Determined List