James Allan

Christopher Aromando

Robert Betlesky

**Table of Contents**

**1.** **EXECUTIVE SUMMARY**

1.1 Project Overview

1.2 Purpose and Scope of this Specification

**2.** **PRODUCT/SERVICE DESCRIPTION**

2.1 Product Context

2.2 User Characteristics

2.3 Assumptions

2.4 Constraints

2.5 Dependencies

**3.** **REQUIREMENTS**

3.1 Functional Requirements

3.2 Security

*3.2.1* *Protection*

*3.2.2* *Authorization and Authentication*

3.3 Portability

**4.** **REQUIREMENTS CONFIRMATION/STAKEHOLDER SIGN-OFF**

**5.** **SYSTEM DESIGN**

5.1 Algorithm

5.2 System Flow

5.3 Software

5.4 Hardware

5.5 Test Plan

5.6 Task List/Gantt Chart

5.7 Staffing Plan

# 

# 

# 

# 1. Executive Summary

## ***1.1*** ***Project Overview***

The first test for the Software engineering triathlon is the endurance test. Main goals of this first endurance test was to collaboratively design, form, solve and organize a successful software code using our robot. Using Sphero Edu, our group's purpose was to generate a code where our robot starts with a green light and says ‘ready set go’ and stops with a red light and says “I’m done and I need water”. As our robot performs, it was critical that our robot followed a well-defined path where it did not collide with any objects, complete a full lap around the track and finally return to the starting path. Our group planned to present this to our instructor and other classmates and be able to answer any concerns or questions. The rest of this document will cover all aspects and steps our group took to successfully complete the first trial and provide other background insight.

## ***1.2*** ***Purpose and Scope of this Specification***

Our purpose was to successfully complete the first trial and be as accurate as possible. Due to unforeseen circumstances, our intended audience will be different than described in the project instructions. Our in scope specifications included a test by one of our group members on the rug after the code was generated. The robot performed to the best of its ability, but often swayed back and forth or wasn’t tight on turns due to the uneven surface. Therefore, we strongly believe our robot would’ve performed 100% accurate if we were in the classroom. Limited space, lack of solid surface and collaborating online are all aspects that we could not control and round up our out of scope specifications.

# 2. Product/Service Description

Overall, the Sphero Edu performed extremely well and was simple to use. We experienced no issues while holding the robot in our possession. The product was easily accessible due to its bluetooth connection, very durable and had no technical malfunctions. Not being able to test the robot in the classroom severely affected aspects and requirements of the project. However, the robot's functionality was never an issue. The robot has many unique features such as the ability to change color and speak out loud. Overall, the product was definitely user-friendly and there would have been no issues if the robot was tested in the classroom.

## ***2.1*** ***Product Context***

No members in our group had previous experience with any product in this market. However, upon research Sphero’s product seems to be the most popular. For example, we googled “Classroom Robot” and Sphero’s were the first images. This product is independent and self contained in some aspects. Once connected to bluetooth and a code is generated, the user may start the program and the robot will go in motion independently. We are not sure if competitors products need user interaction throughout every phase, but we agreed Sphero was a great product for this project. Accessing the product was also extremely simple and very few software or hardware was required. All a user needed was a bluetooth connection and to download Sphero Edu. Based on experience with Sphero’s product, we believe other products would not perform as well as Sphero’s.

## ***2.2*** ***User Characteristics***

We could definitely see this product being used for a variety of courses to perform different objectives due to its accuracy and accessibility. Therefore, our main users are students and faculty. The experience between students who used the product definitely varies. Since we used the robot in a 100 level course, some students had no software programming background or came from completely different majors. The product's simplicity definitely allowed for this to be possible because there was no intimidation aspect. In our opinion, minimal expertise is required and the main characteristic a user needs is patience and willingness to learn.

## ***2.3*** ***Assumptions***

Some assumptions that might affect the requirements include equipment availability (we assume it’d be easy to meet in class and test, but we cannot hand over the robot to others and let other group members test since quarantine), testing space (we assume that we might have space to test in class, but since quarantine, we all might not have enough space at home to test the robot), a specific operating system is assumed to be available; if the operating system is not available, the Requirements Specification would then have to change accordingly.

## ***2.4*** ***Constraints***

Any items that may constrain the design options include operation on an old computer or phone system, system resource constraints (e.g., limits on disk space or other hardware limitations), testing space (some tests require a lot of space which we just do not have), and other design constraints (e.g., design or other standards, such as programming language or framework).

## ***2.5*** ***Dependencies***

List dependencies that affect the requirements.

- Algorithm has to be completed before anything else

- Sphero code has to be written before Sphero code testing can be done

- Sprint documentation is taken after the Sphero code testing

- Sphero robot testing can be done after all the code has been written and tested

- Sphero code adjustments can be adjusted after the Sphero robot testing

- Sprint documentation has to be updated after Sphero code adjustments

# 3. Requirements

Description of system requirements:

* The system that is operating the program must be a system that utilizes the Sphero Edu application and has bluetooth capability

· Describe every input into the system, every output from the system, and every function performed by the system in response to an input or in support of an output. (Specify what functions are to be performed on what data to produce what results at what location for whom.

Functions that need to be performed:

· The algorithm and the steps must be written first before being properly implemented into the Sphero Application to let us know what steps we need to do in order to achieve our objective overall.

· The algorithm and the steps can then be put into the Sphero Application to see how the steps can line up with each process or action given to the robot.

· The Sphero Code must be written in a complete and comprehensive manner to ensure that the code written does not have errors and runs with ease.

· The Sphero Code should be tested (but is not possible due to our certain situation) to ensure that it runs smoothly physically.

· The Sphero Code must then be reviewed and revised for adjustments to ensure that when we choose to modify the code, we can do so properly.

· **Priority Definitions**

The following definitions are intended as the real requirements with proper prioritization.

· Priority 1 – This requirement is regarding functionality and the process of writing the algorithm and is “extremely important” to be completed before other objectives.

· Priority 2 – This requirement is regarding functionality and the process of writing the algorithm to code and has “very strong importance” to be completed before other objectives.

· Priority 3 – This requirement is about reviewing completed algorithm code and is of “strong importance” to be completed to ensure it is done correctly or properly.

· Priority 4 – The requirement is about testing sphero code which is “nice to have or somewhat low importance” because it would help if we could test, but given our situation regarding the quarantine, it is not possible

## ***3.1*** ***Functional Requirements***

In the example below, the requirement numbering has a scheme - BR\_LR\_0## (BR for Business Requirement, LR for Labor Relations). For small projects simply BR-## would suffice. Keep in mind that if no prefix is used, the traceability matrix may be difficult to create (e.g., no differentiation between '02' as a business requirement vs. a test case)

The following table is an example format for requirements. Choose whatever format works best for your project.

For Example:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req#** | **Requirement** | **Description** | **Priority** | **Date Rvwd** | **SME Reviewed / Approved** |
| ENDUR\_01 | Functionality - Algorithm | Writing Algorithm Steps | High | 3/30 |  |
| ENDUR\_02 | Required - Review | Review of Algorithm Steps | High | 3/30 |  |
| ENDUR\_03 | Functionality - Sphero Code | Development of Sphero Code | High | 3/31 |  |
| ENDUR\_04 | Required - Test | Test Sphero Code | Medium | 3/31 |  |
| ENDUR\_XX |  |  |  |  |  |

## ***3.2*** ***Security***

### **3.2.1** **Protection**

Some of the factors that will protect the system from malicious or accidental access, modification, disclosure, or misuse. For example, encryption, activity logging historical data sets, restrictions on intermodule communications, data integrity checks.

### **3.2.2** **Authorization and Authentication**

The role of the Pubcookie login server is that of the trusted, central authentication service. It interacts directly with users. It verifies usernames and passwords with backend authentication services. It issues cookies to users to provide single sign-on functionality and to application servers to provide authentication information.

The role of the Pubcookie application server is that of authentication enforcer. It redirects users who haven’t been authenticated to the login server. It verifies authentication information returned from the login server. It issues cookies to users to maintain authenticated application sessions and provides user authentication information to applications.

## ***3.3*** ***Portability***

Attributes of the system that can help relate to the ease of porting the system to other host machines and/or operating systems include percentage of components with host-dependent code, percentage of code that is host dependent, use of a proven portable language, use of a particular compiler or language subset, use of a particular operating system, the need for environment-independence - the product must operate the same regardless of operating systems, networks, development or production environments.

# 4. Requirements Confirmation/Stakeholder sign-off

Include documentation of the approval or confirmation of the requirements here. For example:

|  |  |  |
| --- | --- | --- |
| **Meeting Date** | **Attendees (name and role)** | **Comments** |
| 03/06/20 | James A., Robert B., Chris A. | Became familiar with Sphero software and assigned different responsibilities for our materials |
| 03/29/20 | James A., Robert B., Chris A. | Assembled Sphero code via VOIP and worked on Sprint 1 Document. |

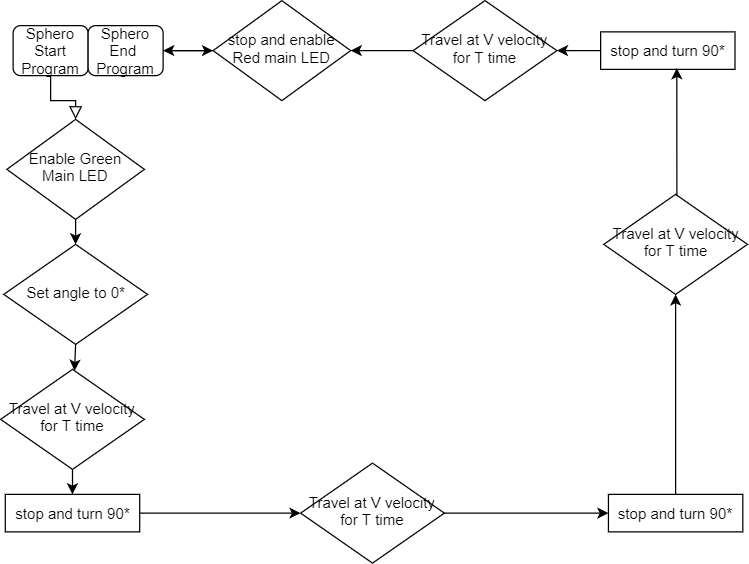
# 5. System Design

This section will provide all details concerning the technical design, staffing, coding, and testing the system

## ***5.1*** ***Algorithm***

* Sphero will begin program
  + Enable green main LED and say “Ready set go”
  + Sphero will go forward in first direction (0°) @ *V* speed
  + Sphero will stop and turn 90° after traveling for *T* seconds
  + Sphero will turn to 180° and continue @ *V* speed for *T* seconds
  + Sphero will stop and turn to the 270° and travel at V speed for T seconds
  + Sphero will stop and turn to the 360° and travel at V speed for T seconds
  + Sphero will enable red main LED and say “I’m done and I need water”
* Sphero end program

## ***5.2*** ***System Flow***



## ***5.3*** ***Software***

Required Software:

* PC Operating System/Iphone iOS
* The SpheroBot uses block coding software available on the Sphero.Edu application.

## ***5.4*** ***Hardware***

Required Hardware:

* SpheroBot
* Computer/Phone (Bluetooth Compatible)

## ***5.5*** ***Test Plan***

Include a test plan showing all unit tests performed for this application, Include test rational, test date, staff member, pass/fail status

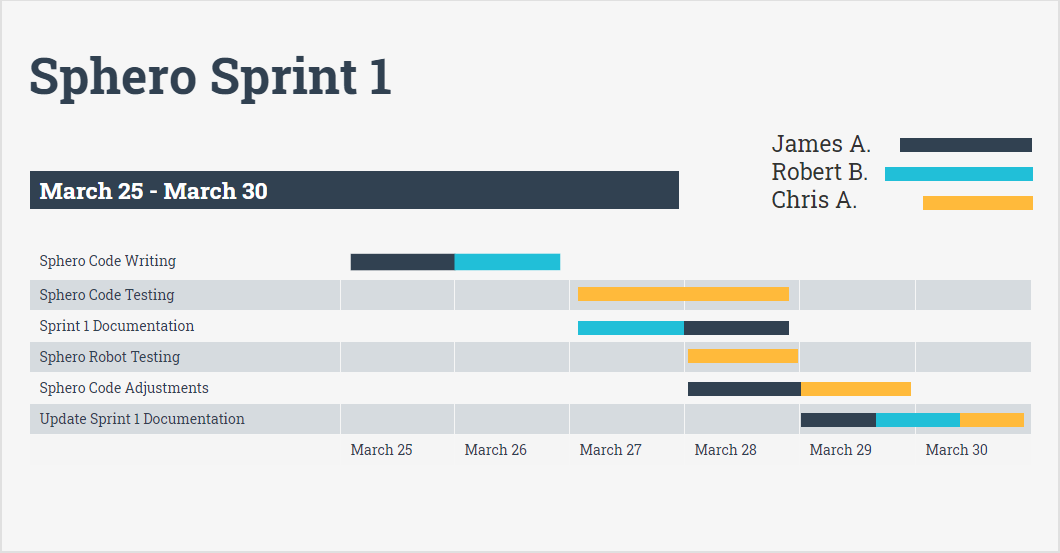
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Reason for Test Case** | **Test Date** | **Expected Output** | **Observed Output** | **Staff Name** | **Pass/Fail** |
| Turn Accuracy | 3/30 | 90\* right angle turn | Was about 2 degrees off | CA | Fail |
| Turn Accuracy | 3/30 | 90\* right angle turn (After adjusting code from 90\* to 88\*) | Considerably more accurate to forming a right angle turn | CA | Pass |
| Time gauge | 3/30 | The robot will need to travel at 100 speed for 3.5 seconds | Robot needed only/about 2.8-3 seconds to make it to the end at 100 speed | CA | Fail |
| Complete Lap | 3/30 | The robot will make a complete rectangular lap, finishing off in about the same area as it started | The robot finished very close to the starting point | CA | Pass |
|  |  |  |  |  |  |

## 

## 

## 

## ***5.6*** ***Task List/Gantt Chart***



## ***5.7*** ***Staffing Plan***

Insert a chart/table that depicts the roles and responsibilities of each team member that worked on this project

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Role | Responsibility | Reports To |
| James A. | Coding/Documentation | Develops shero code to be tested and inputs documentation. | Robert B. |
| Chris A. | Testing/Coding | Tests sphero code with robot and adjusts accordingly. | Robert B. |
| Robert B. | Documentation/Group Management | Inputs documentation and manages group time on different aspects. | Professor Gil Eckert |