System Requirements Specification

for

Overtake and Collision Avoidance with Thymio

Version 0.1

Prepared by Lorenzo Bartolini, Marco Agatensi

University of Florence

02/12/2024

(a modified version of IEEE Software Requirements Specification Template, K.E. Wiegers, 1999)

Table of Contents

1. Introduction 1

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 1

2. Overall Description 2

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 2

2.5 Design and Implementation Constraints 2

2.6 Assumptions of Use 3

3. System Use Cases 3

3.1 Reach the end of the road 3

3.2 Come back to the start 4

4. System requirements definition 5

4.1 Viewpoint 1 or System Feature 1 5

4.2 System Feature 2 (and so on) 5

5. Traceability matrix 6

6. Test Cases/Test Plan 6

6.1 Test Cases/Test Plan Traceability Matrix 6

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Person’s Name, not “Company X”. Multiple people are okay if they all worked on this version |  |  |  |
|  |  |  |  |

# Introduction

## Purpose

<Identify the product whose requirements are specified in this document, including the revision or release number. Describe the scope of the product that is covered by this SRS. If this SRS describes only part of the system or a single subsystem, this should be specified together with the role of this part within the overarching system.>

<Should be very short and to the point.>

## Document Conventions

<Describe any standards or typographical conventions that were followed when writing this SRS, such as fonts or highlighting that have special significance. For example, state whether priorities for higher-level requirements are assumed to be inherited by detailed requirements, or whether every requirement statement is to have its own priority.>

<Optional section>

## Intended Audience and Reading Suggestions

<Describe the different types of reader that the document is intended for, such as developers, project managers, marketing staff, users, testers, and documentation writers. Describe what the rest of this SRS contains and how it is organized. Suggest a sequence for reading the document, beginning with the overview sections and proceeding through the sections that are most pertinent to each reader type.>

<Optional section>

## Product Scope

<Provide a **short** description of the system being specified and its purpose, including relevant benefits, objectives, and goals. If the system being described here is part of an overarching system, its role in the overarching system should be reported.

Relate the system to corporate goals or business strategies. If a separate vision and scope document is available, refer to it rather than duplicating its contents here. >

## References

<List any other documents or Web addresses to which this SRS refers. These may include user interface style guides, contracts, standards, other requirements specifications, use case documents, or a vision and scope document. Provide enough information so that the reader could access a copy of each reference, including title, author, version number, date, and source or location.>

<If nothing is of interest, specify None – but do not remove this section>

# Overall Description

## Product Perspective

<Describe the context and origin of the product being specified in this SRS. For example, state whether this product is a follow-on member of a product family, a replacement for certain existing systems, or a new, self-contained product. If the SRS defines a component of a larger system, relate the requirements of the larger system to the functionality of this software and identify interfaces between the two. A simple diagram that shows the major components of the overall system, subsystem interconnections, and external interfaces can be helpful.>

## Product Functions

<Summarize the major functions the product must perform or must let the user perform. Details will be provided in Section 3, so only a high level summary (such as a bullet list) is needed here. Organize the functions to make them understandable to any reader of the SRS. A picture of the major groups of related requirements and how they relate, such as a top level data flow diagram or object class diagram, is often effective.>

## User Classes and Characteristics

<Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the most important user classes for this product from those who are less important to satisfy.>

## Operating Environment

<Describe the environment in which the system will operate. This includes the hardware platform, operating system and versions, as well as any other cyber or physical components or applications with which it must peacefully coexist.>

## Design and Implementation Constraints

<Describe items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions and standards.>

## Assumptions of Use

<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include legacy or third-party components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as components that you intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).>

# System Use Cases

<The overall use case diagram(s) should be here. Draw use case diagram showing all major features. A rather simple example is given here.>

<The text description of each use case should follow, as in the following sections.>

# 

## Reach the end of the road

1. **Objective** – The two Thymios should be able to reach the end of the road, Thymio A needs to overtake Thymio B without crashing.
2. **Priority** – High
3. **Actors** – Two Thymios (A and B)
4. **Flow of Events** 
   1. **Basic Flow**
      1. Thymio A and B move forward
      2. Thymio A reaches B
      3. Thymio A performs the overtake returning on the road
      4. Thymio A and B move forward
   2. **Alternative Flow(s)**
      1. Thymio A and B move forward
      2. Thymio A reaches B
      3. Thymio A while performing the overtake is not able to return on the road in front of B
      4. Thymio A retries to perform the overtake
   3. **Exception Flow(s)**
      1. Thymio A and B move forward
      2. Thymio A reaches B
      3. Thymio A crashes on B while performing the overtake
5. **Includes** - none
6. **Preconditions** – Thymio A is behind Thymio B, Thymio A is faster than Thymio B
7. **Post conditions** – Thymio A reach the end of the road, Thymio B still needs to reach the end of the road
8. **Notes/Issues** - none

## Arrive at destination

1. **Objective** – The two Thymios should be able to arrive at destination without crashing when doing it
2. **Priority** – High
3. **Actors** – Two Thymios (A and B)
4. **Flow of Events** 
   1. **Basic Flow**
      1. Thymio A turns around while B moves forward to reach the end of the road
      2. Thymio A and B face eachother going in opposite directions
      3. They avoid the collision
      4. Thymio A and B continue on the road
      5. Thymio B reaches the end
      6. Thymio A and B arrive at destination
   2. **Alternative Flow(s)**
      1. TODO
   3. **Exception Flow(s)**
      1. TODO
5. **Includes** - none
6. **Preconditions** – Thymio A is at the end of the road, Thymio B still needs to reach the end of the road
7. **Post conditions** – Thymio A and B has arrived at destination
8. **Notes/Issues** – The destination of A is the starting position, the destination of B is the end of the road

# System requirements definition

<Organize the requirements for the product by system features.

Following the structure of the course, we recommend to organize this section using viewpoints. For example, providing subsections containing requirements of the:

* Architecture viewpoint
* Interface and Communication viewpoints
* Time and time synchronization viewpoint
* …

## Viewpoint 1 or System Feature 1

<Present the detailed requirements associated. Requirements should be concise, complete, unambiguous, verifiable, and necessary. >

<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind. That you have created in for use cases. >

<To add information that is not part of the requirement, for example explanations, **Notes** can be added. Remember that Notes are not requirements and will not be validated or further traced through the lifecycle.>

REQ-Viewpoint1-1:

REQ Viewpoint1-2:

…

## System Feature 2 (and so on)

# Traceability matrix

<If an upper layer of documentation is available, it is necessary to include traceability of requirements. >

|  |  |
| --- | --- |
| **Requirements from upper layer document (identify precisely the document and its version)** | **This SRS** |
| [Req-1] | [Req-Viewpoint1-1], [Req-Viewpoint1-2], … |
| … | … |
|  |  |
|  |  |
|  |  |
|  |  |

# Test Cases/Test Plan

<Define test cases for testing your features and requirements.>

<Most often, this is in a separate document. Tests should be defined at this stage to facilitate validation of requirements.>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Scenario** | **Test Steps** | **Test Data** | **Expected Results** |
| TC01 | Withdraw money | - Insert card  - Insert PIN  … | PIN value: correct pin | Money withdrawn |
| TC02 |  |  |  |  |
| TC03 |  |  |  |  |
| … |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

## Test Cases/Test Plan Traceability Matrix

<Test cases must be traced to requirements, to prove that all requirements have been considered for testing, and tests have been developed whenever appropriate.>

<Note that requirements should be testable, so there must be good reasons to not have tests matched to a requirement.>

|  |  |
| --- | --- |
| **SRS Requirements** | **Test Cases** |
| [Req-Viewpoint1-1] | [TC01], [TC02],… |
| … | … |
|  |  |
|  |  |
|  |  |