System Requirements Specification

for

EGR101 Simulation

Version 1.0 approved

Prepared by:   
Emily Connearney  
Luke Crump  
Vivian Dang  
Keely Mashburn  
Daniel Khalil

Embry Riddle Aeronautical University

9/22/21

Table of Contents

Table of Contents ii

Revision History ii

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 User Documentation 2

2.7 Assumptions and Dependencies 3

3. External Interface Requirements 3

3.1 User Interfaces 3

3.2 Hardware Interfaces 3

3.3 Software Interfaces 3

3.4 Communications Interfaces 3

4. System Features 4

4.1 System Feature 1 4

4.2 System Feature 2 (and so on) 4

5. Other Nonfunctional Requirements 4

5.1 Performance Requirements 4

5.2 Safety Requirements 5

5.3 Security Requirements 5

5.4 Software Quality Attributes 5

5.5 Business Rules 5

6. Other Requirements 5

Appendix A: Glossary 5

Appendix B: Analysis Models 5

Appendix C: To Be Determined List 6

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Purpose

This Product is designed to supplement learning for remote students in the course EGR101. The course utilizes a $229.00 Parallax Boe-Bot Robot Kit to allow students to design a functionally autonomous robot. EGR101’s main project deliverables include grades based on performance in four Boe-Bot courses built to challenge students on forming solutions to: basic line following, line following corrected for noise, object avoidance, and resource management. Normally students are split into groups of three, with each group receiving a Boe-Bot kit, which includes its respective sensors, LEDs, and resistors. Due to the recent pandemic, the role of the course has changed due to variability of student in-person attendance. The current solution to this problem was to make students pay $85 for their own kits and perform the required deliverables remotely.

sensors, LEDs, and resistors.

## Document Conventions

This document intends to define all requirements and conditions associated with the EGR101-Simulation System. This document covers the product itself, its interaction with the user, and the requirement associated with bringing the system in compliance with the EGR101 course vision.

## Intended Audience and Reading Suggestions

The intended users of this product include students, and the instructor of the EGR101 course. This document contains information on product functionality, requirements, standards and regulations given by the current EGR101 instructor. Section 2 describes the intent of the project. Section 3 describes the technical aspect most pertinent to the other software developers maintaining or enhancing the project. Section 4 and 5 describe the features and standards the project follows and should be of interest to instructors interested in understanding the scope of this tool.

Along with reading this document, the references outlined in section 1.5 are important to know for the proper operation and testing of the project.

## Product Scope

The scope of this application would be to reduce the cost of eventual replacement of Boe-Bot kits, allow for remote learning through testing electronic based solutions in a sandbox environment, and ease of grading said electronic based solutions. The proposed project would Allow students to program Arduino sketches, design a virtual bot through adding components and wire connections, test their virtual bot on the 4 deliverable courses and provide a sandbox environment to improve understanding of basic circuitry and imperative programming. This product could be used in applications far beyond the scope of this course as a virtual electronics test environment could be in-valuable to autonomous vehicle testing

## References

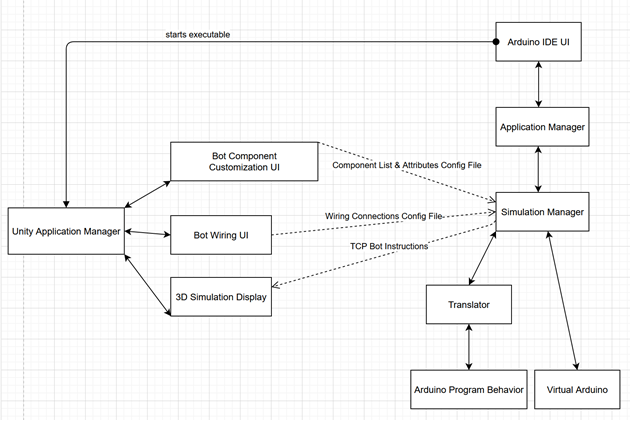
*[1] Language reference*. Arduino Reference - Arduino Reference. (n.d.). Retrieved September 30, 2021, from <https://www.arduino.cc/reference/en/>.

# Overall Description

Section 2 covers purpose of the product in 2.1, the product functions in 2.2 associated user information in 2.3, and the operating environment in 2.4, Sections 2.5, 2.6 and 2.7 contain additional information on the implementation of the functions outlined in section 2.2

## Product Perspective

The EGR101 Simulation System will act as a sandbox to test Arduino code and wiring configurations from the Arduino to its individual components on the simulated bot. The application should be controlled out of the core Java application which will initialize a Unity application displaying 3-Dimensional Bot customization and simulation display. This would put both the core application thread of execution and the Unity thread of execution in parallel. These threads will communicate with each other over TCP.



## Product Functions

2.2.1 Simulate an Arduino IDE to allow users to program virtual bots

2.2.2 Emulate Arduino and allow IDE to compile and change Arduino behavior at runtime

2.2.3 Provide Arduino API that will modify the emulated Arduino.

2.2.4 Allow components to be added to the bot configuration

2.2.5 Allow for wired connections between components and Arduino to be specified.

2.2.6 Simulate component functionality based off wired connections.

2.2.7 Display 3-Dimensional autonomous bot course simulation

2.2.8 Produce recording of 3-Dimensional course simulation

2.2.9 Produce zipped configuration files for wired connections to components, component list, and Arduino sketches

## User Classes and Characteristics

### Student

The student is the default user intended to interact with this system. The student is not expected to understand how the Arduino API functions, or how to properly wire Arduino components. Students will interact with the Arduino IDE to produce code, update the bot configurations through adding/removing components, and wiring the components to power, ground, and IO ports on the Arduino, students can also test their configured bots on the 4 deliverable courses. This application will be mainly focused on providing a user-friendly experience for these students.

### Instructor

The instructor is a secondary user who intends to interact with this system. This individual will most likely have proficient knowledge in the systems this application is emulating. This user will require students to have configuration files exported and imported to help ease of grading. There will also be a system in which students must fill out a profile which will be injected into the configuration file which will give the instructor information about student name, student ID, and any other information needed for the identity of the student.

## Operating Environment

<Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.>

## Design and Implementation Constraints

The software shall execute on Mac OS and Windows OS as a local desktop application. The only necessary software the User will require is access to Java. Design constraints dictate that this application must work with low end hardware as it might be used on school computers.

## User Documentation

<List the user documentation components (such as user manuals, on-line help, and tutorials) that will be delivered along with the software. Identify any known user documentation delivery formats or standards.>

## Assumptions and Dependencies

<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial 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 software 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).>

# External Interface Requirements

## User Interfaces

<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>

## Software Interfaces

<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>

## Communications Interfaces

<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.>

# System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

## Wiring and Design Interface

Section 4.1 defines a list of requirements for the wiring and design interface.

**4.1.1 Description and Priority**

The Wiring and Design Interface will allow the user to design a robot in 3-dimensional space. This would include adding components, mounts, and cosmetic changes. The wiring and design GUI will also allow the user to connect components to the built in Arduino to make functional circuits.

**4.1.2 Stimulus/Response Sequences**

**4.1.3 Functional Requirements**

1. The system shall allow the user to select a wire.
2. The system shall allow the user to move the wire to connect two pin locations.
3. The system shall record the pin locations that are connected by a wire in a file.
4. The system shall have a button to save the current wire configuration.
5. The system shall save the current configuration of wires when the “save” button is pressed.
6. The system shall have a button to exit the Wiring and Design Interface.
7. The system shall exit to the main view screen when the exit button is pressed.
8. The system shall have a button to add a resistor to the board.
9. The system shall be able to connect resistors to pins.
10. The system shall be able to connect resistors to wires.

## Arduino IDE

Section 4.2 defines a list of requirements for the Arduino IDE.

**4.2.1 Description and Priority**

**4.2.2 Stimulus/Response Sequences**

**4.2.3 Functional Requirements**

1. The system shall have a File button that shows file operations when clicked on.
2. The system shall have a save button appear when the File button is clicked
3. The system shall save the Arduino script when the save button is clicked.
4. The system shall have an open button appear when the File Button is clicked.
5. The system shall open a new Arduino script when selected
6. The system shall have a “save configuration” button appear when the File button is clicked.
7. The system shall save a configuration file when the “save configuration” button is clicked.
8. The system shall produce a configuration file that contains the wiring setup and the Arduino code when the “save configuration” button is pressed

## Bot Simulation

Section 4.3 defines a list of requirements for the Bot Simulation.

**4.3.1 Description and Priority**

**4.3.2 Stimulus/Response Sequences**

**4.3.3 Functional Requirements**

1. The system shall prompt the user to select a course, the course selected will be displayed and executed on.The system shall have a save button appear when the File button is clicked
2. The emulation thread shall send each components behavior to the simulation UI via TCP connection per cycle.The system shall have an open button appear when the File Button is clicked.
3. The simulation UI shall send each components sensor data to the Arduino Emulation via TCP connection per update iteration.
4. The system shall have a “save configuration” button appear when the File button is clicked.
5. The simulation UI shall update the Boe-Bot’s rotation if one wheel is rotating
6. The simulation UI shall update the Boe-Bots position if both wheels rotate
7. The system should display console logs in the Arduino IDE during simulation execution if logging is present within Arduino code.
8. The system shall restart the simulation if the restart button is selected
9. The system shall end the simulation if the end button is selected.
10. The system shall pause the simulation if the pause button is selected.
11. The system shall play the simulation if the play button is selected.
12. The system shall not allow the boe-bot to change position if it is colliding with an obstacle in the course.

## Arduino Emulation

Section 4.4 defines a list of requirements for the Arduino Emulation.

**4.4.1 Description and Priority**

**4.4.2 Stimulus/Response Sequences**

**4.4.3 Functional Requirements**

1. The system shall allow for pin connections between the Arduino pins and components
2. The system shall simulate the Arduino clock by counting in microseconds after executing
3. The system shall allow for calls from Arduino code to delay programmable interaction with the components
4. The system shall match the behavior defined from compiled Arduino code
5. The system shall allow for digital writing to pins by providing voltage via simulated Pulse Width Modulation (PWM)
6. The system shall allow for analog writing to pins by providing a voltage
7. The system shall provide traditionally used libraries like Servo.h and Serial.h to interface with.
8. On execution the system shall push voltage to each pin sequentially beginning at the IO and power ports of the Arduino.
9. On execution the components shall work if and only if they have adequate ground connection, and enough voltage to satisfy the potential of the component.

## System Feature 2 (and so on)

# Other Nonfunctional Requirements

## Performance Requirements

<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>

## Safety Requirements

<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product’s design or use. Define any safety certifications that must be satisfied.>

## Security Requirements

<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>

## Software Quality Attributes

<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>

## Business Rules

<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>

# Other Requirements

<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>

Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

Appendix C: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>