Development Process and Implementation

Group 4: Autonomous Foosball Table

Revision 0

October 18, 2015

Members:

|  |  |
| --- | --- |
| Yue(Ryan) Ma | 1147335 |
| Chenhe Li | 1158035 |
| Taha Hussain | 1200349 |
| Viktor Smirnov | 1154392 |
| Zi(Roland) Zhou | 1216063 |
| Alvin Li | 0957972 |

# 

# 

# 1 REVISIONS

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Description |
| 0 | October 19, 2015 | Yue(Ryan) Ma,  Chenhe Li,  Taha Hussain,  Viktor Smirnov,  Zi(Roland) Zhou,  Alvin Li | Initial version |

# 2 TABLE OF CONTENTS

|  |  |
| --- | --- |
| [Revisions](#h.l1f9bdxm31dh) | 1 |
| [Table of Contents](#h.pnsstzv5r62n) | 2 |
| [Introduction](#h.ifjt5cmja0un) | 4 |
| [Purpose](#h.lgvch19ebiu4) | 4 |
| [Scope](#h.9fzedsmywgwe) | 4 |
| [References](#h.wj9v4wzc4jl3) | 4 |
| [Development process](#h.wi9eyx56k256) | 5 |
| [Overall development process](#h.ubsiilrddjv9) | 5 |
| [Detailed development process](#h.9wysgy5mng3s) | 7 |
| [Roles and responsibilities](#h.5jv1nrllk2cv) | 10 |
| [Tools and standard](#h.golaq2h7glxf) | 11 |
| [Tools](#h.apww76vql5ad) | 11 |
| [Standard](#h.6jqc4enrf06y) | 11 |
| [Version control](#h.x0d8dtk03h3r) | 12 |
| [Tool](#h.5p3c6tv0izzx) | 12 |
| [Version controlled item](#h.rofxnne58uv2) | 12 |
| [Software:](#h.uxex3uny4pva) | 12 |
| [Electrical:](#h.tf73qfvqvhai) | 12 |
| [Mechanical:](#h.d83rgnx8next) | 12 |
| [Documentation:](#h.rvrvj9usguuy) | 12 |
| [Usage of version control](#h.3skwlm2r32du) | 13 |
| [General](#h.yanrdxk7z0g) | 13 |
| [Bug report and tracking](#h.5gpk6z4nj8gr) | 13 |
| [Change request](#h.c849wom0ohx1) | 13 |
| [Bug and change documentation](#h.2ukvg78f7yed) | 13 |
| [Issue labeling](#h.b56yendqd37p) | 13 |

# 3 INTRODUCTION

## 3.1 Purpose

The purpose of this Development Process and Implementation document is to define the overall develop process flow and how each step should be implemented. This document will be used as an execution plan for the development autonomous foosball table and help teaching assistants and supervising professor as well as ourselves in tracking our development progress.

## 

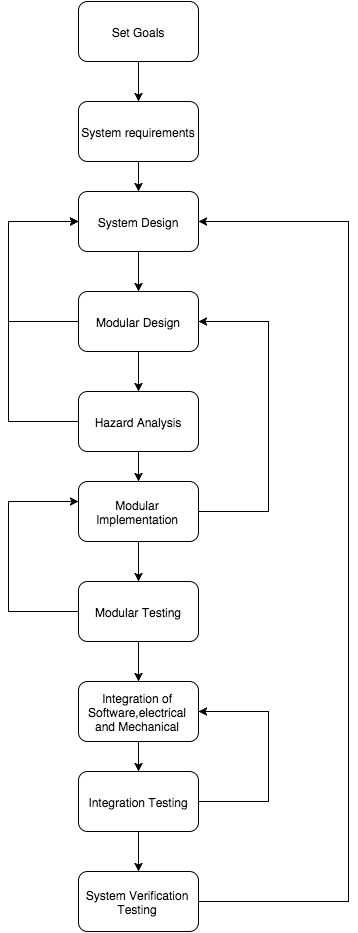
## 3.2 Scope

The focus of this document is on the steps that needed to be done during the overall development, including roles and responsibilities of team members, tools and versions of tools will be used and method of version control.

# 4 DEVELOPMENT PROCESS

## 4.1 Overall development process

Development process is based on V-model process. (see Figure 1.1 below)

 Figure 1.1

## 4.2 Detailed development process

1. Set projects goals
   1. Input:
      1. List of official rules for foosball
      2. Statistics of the ability of an average player
      3. Possible customer research
      4. existing products
   2. Output:
      1. A list of actions and rules that must be achieved and followed by a robotic foosball table that simulates a human player and provides competitive gameplay
   3. Acceptance Criteria:
      1. Goal are achievable
      2. Goals should be clear, concise and well defined.
      3. Goals should have reasonable justification
2. System requirements
   1. Input:
      1. A list of actions and laws that must be achieved by a robotic foosball table that simulates a human player
   2. Output:
      1. Functional requirements: a list of actions required to achieve the goals
      2. Non-functional requirements: a list of metrics required for effectiveness measurement of the functional requirements
      3. Controlling and monitoring variables
      4. Functional Decomposition Diagram
   3. Acceptance Criteria:
      1. Requirements should not contradict each other
      2. Requirements should be concise and cover all possible actions and reactions of a system
      3. Requirements should be aimed at fulfilling the goals
      4. There should be no redefinitions of requirements
3. System design
   1. Input:
      1. functional requirements
      2. non-functional requirements
      3. functional decomposition diagrams
   2. Output:
      1. a workflow of interaction between the subsystems required to achieve the functional requirements
      2. Behaviour description for all the components
   3. Acceptance Criteria:
      1. The system should meet the requirements
      2. The design should have its input and outputs well defined and the output of the entire system should be in compliance with the goals desired from the system
      3. Each module should not affect other module’s behaviour in any other way than intended in the design
4. Modular design: software design/mechanical design/electrical design
   1. Input:
      1. A workflow of interaction between the subsystems required to achieve the functional requirements
      2. Non-functional requirements
      3. Cost analysis for parts
      4. Behaviour description of all the components
   2. Output:
      1. Interface specification for each module
      2. Internal design for each module
      3. A simulation, prototype or engineering drawing of all subsystems
   3. Acceptance Criteria:
      1. The input and outputs of subsystems are restricted the pairs of input and output to reflect all requirements
      2. All the routine satisfying the input/output relation is correct and acceptable
      3. Each module should meet the requirements for this particular module
5. Hazard analysis
   1. Input:
      1. Workflow of all subsystems
      2. Functional requirements
      3. Simulation/prototype of all major subsystems
   2. Output:
      1. Flowchart of system functions
      2. FMEA Chart
   3. Acceptance Criteria:
      1. Should prove that the analysis covers all possible outcomes of the system with all possible inputs
      2. Clearly defines which standard was used and its justification and how that standard was followed during the analysis
6. Modular implementation: software implementation/mechanical implementation/electrical implementation
   1. Input:
      1. Modular design for each module
      2. An interface specification for each module
   2. Output:
      1. Working subsystems
   3. Acceptance Criteria:
      1. Each module meets the modular internal design requirement.
      2. The input and outputs of each module meet the interface specification
7. Modular testing: software testing/electrical testing
   1. Input:
      1. Working subsystems
      2. System requirement
   2. Output:
      1. Verified subsystems
      2. Test log
   3. Acceptance Criteria:
      1. All the cases of each module are tested
      2. The expected output and the output of the testing module are the same for the same testing input

1. Integration of software, electrical and mechanical system.
   1. Input:
      1. Working subsystems
      2. System requirement
   2. Output:
      1. Integrated system
   3. Acceptance Criteria:
      1. Integrated system meets the requirement
2. Integration testing
   1. Input:
      1. Integrated system
      2. System requirement
   2. Output:
      1. A working system
      2. Test log
   3. Acceptance Criteria:
      1. All the cases of each module are tested
      2. The expected output and the output of the integrated system are the same for the same testing input
3. System verification testing
   1. Input:
      1. A working system
      2. Requirement
      3. Project goal
   2. Output:
      1. A list of metrics that state if the requirements are met
      2. Test log
   3. Acceptance Criteria:
      1. The behaviour of the system match the goals of the project
      2. The behaviours of the system satisfy the requirements specification.
      3. The outputs from black-box and white-box tests are the same as the expected values

## 4.3 Roles and responsibilities

This project will be divided into three major component: software, electrical and mechanical. Table 1.1 displays the responsibilities and work distribution within the team. Table 1.2 states the leader for individual component.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Software | Electrical | Mechanical |
| Ryan Ma | development/testing |  |  |
| Chenhe Li | development |  |  |
| Taha Hussain | development | development/testing |  |
| Viktor Smirnov |  | development | development/testing |
| Roland Zhou | development/testing |  |  |
| Alvin Li |  | development/testing | development |

Table 1.1 Responsibilities of team members

|  |  |  |  |
| --- | --- | --- | --- |
| Software lead | Electrical lead | Mechanical lead | Documentation lead |
| Chenhe Li | Taha Hussain | Viktor Smirnov | Ryan Ma |

Table 1.2 Component leaders

# 5 TOOLS AND STANDARD

## 5.1 Tools

* Software:
  + Language: C++ 11, C
  + IDE: Visual Studio 2015 Community, Keil uVision 4.0.0
  + External library: OpenCV 3.0 for C++
  + CAD tool: AutoCAD 2015
  + Algorithm simulation: Matlab 2014a
  + Circuit design and simulation: Multisim Student Edition
  + PCB layout: KiCad 4.0.0 RC1
* Hardware:
  + Microcontroller: STM32F4 Discovery board

## 5.2 Standard

* Coding standard: Google C++ Coding Standard

# 

# 

# 6 VERSION CONTROL

## 6.1 Tool

A private Github repository will be used as our version control tool. Git and Github are known to be easy to learn and use. Furthermore Git is integrated with most popular IDEs which makes version control more effective.

## 6.2 Version controlled item

### 6.2.1 Software:

* Source code of the user program including config files
* External libraries used in the project
* User program(binary file)

### 6.2.2 Electrical:

* Circuit designs:
  + Transfer functions
  + Schematic
  + Truth tables
  + Electronic components and ratings
  + CAD designs
* Circuit simulations of circuit designs

### 6.2.3 Mechanical:

* Hardware dimensions (length, mass, diameter, and etc.)
* CAD modeling

### 6.2.4 Documentation:

* Development process & implementation
* Project goals
* Requirement
* System design
* Hazard Analysis
* Implementation
* V & V

## 6.3 Usage of version control

### 6.3.1 General

For software, one member will create an empty base project and the others will synchronize this base project to local. Ideally each member should work on different modules at the same time to reduce the number of conflicts when pushing the files to the repository. When committing sources files, the team member shall state the changes in the file and reasons for the changes in the commit comment. In software component team members need to make sure the committed changes will not break the current build. Also members shall always pull from the repository to get the updated version of the project before modifying any source file.

### 6.3.2 Bug report and tracking

A bug report will be opened using Github’s “issue” feature when a bug is found by the tester with label “bug”. In the Github issue, the tester shall keep a record of the test case failed and logs about the failure. Then the component leader shall assign the bug to appropriate member to fix. Once the bug is fixed, tester must validate the fix first before closing the issue.

### 6.3.3 Change request

Similar to bug report, change request shall also be opened by Github’s “issue” feature. If a team member decides to make a change from the design level, a change request must be opened in the issue with label “changerequest”. Component leaders shall then review the request and make the decision. If the request is approved, component leader then assigns the request to appropriate team member.

### 6.3.4 Bug and change documentation

Bugs and changes shall be documented by Github’s “issue” feature. Proper description must be added in the comment when opening and closing the issue.

### 6.3.5 Issue labeling

The following labels must be including when opening a new issue. The labels can classify issues thus significantly increases project management proficiency.

* bug: issue is a bug report
* changerequest: issue is a change request
* enhancement: issue is about enhancement