Project Title: System Verification and Validation Plan for MECHTRON 4TB6

Team 25, Formulate
Ahmed Nazir, nazira1
Stephen Oh, ohs9
Muhanad Sada, sadam
Tioluwalayomi Babayeju, babayejt

October 31, 2022

1 Revision History

Date	Developer	Notes
October 30	Stephen	Added General Information and Plan
Date 2	1.1	Notes

Contents

List of Tables

List of Figures

2 Symbols, Abbreviations and Acronyms

symbol	**description**
T	Test

This document ...

3 General Information

3.1 Summary

Formulate consists of four subsystems, one hardware subsystem and three software subsystems, which interact to provide the user with a testing device designed to eliminate automatable processes in common testing procedures.

A physical data collection device is the hardware subsystem used as the first point of contact with the measured quantity through a sensor. The sensor obtains physical quantities for the device to buffer, before sending the data to a desktop application for user verification.

The user has the ability to view the data collected by the physical device after a completed test using a desktop application software subsystem. The desktop application enables the user to either accept the test results to then store a collection of data from a test to a database, or reject the test and prevent the test from being stored to a database.

Accepted test data sent from the desktop application aggregates and saves verified test results to a database software subsystem. Users can query the database to obtain common statistics in test data and generate new or obscure relationships by leveraging database language capabilities.

A final dashboard software subsystem then queries the database to visualize key performance indicators on the test data collected and stored in the database.

3.2 Objectives

The objective of the system Verification and Validation (VnV) plan for Formulate is to ensure the intended project qualities are present.

Ease in user understanding is a quality Formulate will achieve to support the system's usability. Specifically, ease in user understanding of each subsystem's function and how subsystems interface will be key qualities of the overall project.

The system will also demonstrate the quality of adequate portability and physical robustness to support system maintainability, portability, and operationality.

3.3 Relevant Documentation

Talk about how this document draws from the system requirements gathered during software requirements specification (SRS) and hazard analysis.

This document references a variety of requirements generated during the Software Requirements Specification (SRS) process and the Hazard Analysis (HA) process for the Formulate system.

Author (2019)

4 Plan

4.1 Roadmap

The intention of testing for Formulate is to generate confidence that the project meets qualities relating to usability, maintainability, portability, operationality, and safety set out as requirements in SRS and HA documentation. Through sets of unit and system tests that will prove if the system has met the above requirements, Formulate will understand if the project has achieved the desired qualities.

Specifically, requirements that are functional, non-functional, and safety-security related from the SRS and HA documents will be referenced in the Plan, System Test Description, and Unit Test Description sections of this document.

4.2 Verification and Validation Team

Name	Role	Explanation
Stephen	Desktop Application Tester	VnV for software application design and integration with hardware and database
Ahmed	Hardware Device Tester	VnV for embedded program design, chassis design, and integration with desktop application
Muhanad	Database Application Tester	VnV for database design and integration with desktop application and dashboard
Tioluwalayomi	Dashboard Application Tester	VnV for dashboard application design and integration with database
Timofey	Project and Course Teaching Assistant	Detailed low level feedback on planned VnV tests
Dr. Smith	Course Instructor	General high level feedback on planned VnV tests

4.3 SRS Verification Plan

SRS verification will be composed of two approaches to verify that functional and non-functional requirements are met. The first approach is engaging in read through's of the SRS document each month. Individual member progress will be evaluated against the relevenat sections(s) of the SRS document to ensure system development is on track to meet the requirements. The second approach is evaluating issues created by classmates on GitHub and incorporating their concerns and suggestions as seen fit.

Stephen and Tioluwalayomi will lead the group wide discussion for SRS verification activities on the first Tuesday of each month.

4.4 Design Verification Plan

Design verification will be composed of two approaches. The first planned approach is completing read through's of each individual's high level design documentation for their respective sub-system. The design documentation covered during these read through's will likely entail mechanical, electrical, and or software schematics or diagrams outlining the architecture of the subsystem. Members will voice concerns during the read through of design decisions made by the individual responsible for the sub-system architecture. The second planned approach is evaluating issues created by classmates on GitHub and incorporating their concerns and suggestions as seen fit.

Ahmed will lead the group wide discussion for design verification activities on the second Tuesday of each month.

4.5 Implementation Verification Plan

Implementation verification will be composed of techniques in both static and dynamic analysis.

Content walkthrough is the primary type of static implementation verification technique the group plans on using. Three similar types of content walkthrough's are planned for use depending on the sub-system under analysis. Software implementation's will receive a code walkthrough, mechanical implementation's will receive a Computer Aided Design (CAD) spin, and electrical implementation's will receive a schematic walkthrough. During the content walkthrough, unit and system tests relevant to the implementation will be considered to critique the quality of the implementation. A meeting will be organized for each content walkthrough once the implementation has reached a notable milestone worthwile for group analysis.

Live execution of the implementation using a proof of concept style demonstration to the group is the primary type of dynamic implementation technique the group plans on using. During the live demonstration of the implementation, system and unit tests relevant to the implementation will be considered to critique the quality of the implementation. Using the initial state and inputs of the test outlined in the system and unit test sections, the quality of implementation is passed or failed depending on if the actual output of the implementation matches the expeted output.

Ahmed and Muhanad will lead the group wide discussion for design verification activities on the second Tuesday of each month.

4.6 Automated Testing and Verification Tools

Tools will not be used to automate testing, profile, or code coverage for software flashed on embedded hardware because of the available tools incur high additional overhead to testing effort and times. The group plans on completing extensive unit testing for embedded software to compensate for the absence of automated testing and coverage tools.

The desktop application will likely use Visual Studio's memory usage tool to profile the program during execution.

PyLint and SQLFluff will be used as the static code analysis tools to support uniformity in the desktop application and database programs respectively.

Code coverage will be completed manually for the desktop, database, and the dashboard programs. This decision will be feasible as the expected size of software for the applications listed above is relatively small. As a result, it is reasonable for to manually check the amount of code coverage achieved through tests.

4.7 Software Validation Plan

There are no current plans to use external data for validation.

5 System Test Description

5.1 Tests for Functional Requirements

5.1.1 Area of Testing1

Title for Test

1. test-id1

   Control: Manual versus Automatic

   Initial State:

   Input:

   Output:

   Test Case Derivation:

   How test will be performed:

2. test-id2

   Control: Manual versus Automatic

   Initial State:

   Input:

   Output:

   Test Case Derivation:

   How test will be performed:

5.1.2 Area of Testing2

...

5.2 Tests for Nonfunctional Requirements

5.2.1 Performance

Operational in physical environment

1. Operational in physical enviornment

   Type: Dynamic, Manual

   Initial State: Device is on and mounted to the device, has connected to the application and is waiting to start measuring.

   Input/Condition: Vehicle's motor starts and values start to get picked up by device

   Output/Result: Device is operational and stays physically intact in all types of weather and at 20% greater than threshold values.

   How test will be performed: The device will be tested outdoors under various weather conditions including rain, windy, etc. The device will also be tested in temperature and vibration conditions that are above threshold values. This will be performed by placing the device in a hot environment and vigoursly shaking it while being on a stationary mount.

2. Viewing live data

   Type: Dynamic, Manual

   Initial State: Device is on and mounted to the device, has connected to the application and is waiting to start measuring.

   Input: Vehicle's motor starts and values start to get picked up by device

   Output/Result: Data latency should be less than 30 seconds to simulate viewing live data.

   How test will be performed: The amount of time for data to start being viewable on the application will be inspected to be less than 30 seconds. The application will also be inspected to ensure that data is smooth and not lagging while measurements are being performed.

3. Modularity and Maintainability

   Type: Dynamic, Manual

   Initial State: Device is on measuring and sending values to the application, and connection to database has been verified

Input: Either the device, application, or database is disconnected or turned off

Output: The other two components are still functional even though communication between them is broken.

How test will be performed: While device, application, and database are fully functional and communicating successfully, different combinations of either one or two components will be turned off. The other component(s) will be inspected to ensure that they are operational and indicating that the other component(s) are disconnected.

4. test-id2

   Type: Functional, Dynamic, Manual, Static etc.

   Initial State:

   Input:

   Output:

   How test will be performed:

5. test-id2

   Type: Functional, Dynamic, Manual, Static etc.

   Initial State:

   Input:

   Output:

   How test will be performed:

5.2.2 Usability

1. Mounting hardware and starting measurements

   Type: Dynamic, Manual

   Initial State: Device is turned off and nothing is connected, only the application is loaded on to the computer

Input: Users will be asked to setup device and start taking measurements, rate setup process using a survey

Output: Time for setup and data to appear on the application should be less than 5 minutes and

How test will be performed: A test group will be educated on the setup and connection of the device, then they will attempt to do that process. Each person will be timed and compared to the 5 minute threshold. In addition, they will be given a survey to rate the setup process on a scale from 1 to 5 the following categories: ease of use, need for assistance,

2. test-id2

   Type: Dynamic, Manual

   Initial State: Device is given to McMaster's Formula E team to use

   Input: Using a survey, Formula E members will compare their current testing process to the Formulate process

   Output: All users need to select Formulate in at least 2 of the 3 categories

   How test will be performed: Formula E members will select which process is preferred in the following categories: speed, data collection, ease of use

3. test-id2

   Type: Dynamic, Manual

   Initial State:

   Input:

   Output:

   How test will be performed:

5.3 Traceability Between Test Cases and Requirements

6 Unit Test Description

6.1 Unit Testing Scope

6.2 Tests for Functional Requirements

6.2.1 Module 1

1. test-id1

   Type:

   Initial State:

   Input:

   Output:

   Test Case Derivation:

   How test will be performed:

2. test-id2

   Type:

   Initial State:

   Input:

   Output:

   Test Case Derivation:

   How test will be performed:

3. ...

6.2.2 Module 2

...

6.3 Tests for Nonfunctional Requirements

6.3.1 Module ?

1. test-id1

   Type:

   Initial State:

   Input/Condition:

   Output/Result:

   How test will be performed:

2. test-id2

   Type: Functional, Dynamic, Manual, Static etc.

   Initial State:

   Input:

   Output:

   How test will be performed:

6.3.2 Module ?

...

6.4 Traceability Between Test Cases and Modules

References

Author Author. System requirements specification. https://github.com/..., 2019.

7 Appendix

This is where you can place additional information.

7.1 Symbolic Parameters

The definition of the test cases will call for SYMBOLIC_CONSTANTS. Their values are defined in this section for easy maintenance.

7.2 Usability Survey Questions?