Reflection Report
Formulate

Team 25, MECHTRON 4TB6
Ahmed Nazir, nazira1
Stephen Oh, ohs9
Muhanad Sada, sadam
Tioluwalayomi Babayeju, babayejt

April 5, 2023

Contents

1 Changes in Response to Feedback

1.1 SRS and Hazard Analysis

There have been two updates made to the project's requirements and analysis documents. Firstly, a functional decomposition diagram has been added, which breaks down the system into its constituent parts and shows how they are related to each other. This diagram will help to provide a clearer understanding of the system's functionality and how it works.

Secondly, the functional requirements have been updated based on feedback from the Formula E team. This feedback has been used to refine and clarify the requirements to ensure that they are accurate and complete.

1.2 Design and Design Documentation

In the latest update to the design and design documentation, we have made several changes to the system design. Specifically, we have added updated pictures of the graphical user interface (GUI) to provide a more accurate representation of the final product. Additionally, we have included new functions in the description of the ui_functions.py module in the management information system (MIS).

Furthermore, we have introduced a new module called generate_arduino_code in the MIS, which will enable us to automatically generate Arduino code based on the user's inputs. This module will greatly simplify the development process and make it more efficient.

Lastly, we have updated the hardware section to include all of the new components that have been added in the final revision. This will provide a comprehensive overview of all the hardware that is required for the system to function properly. With these updates, we are confident that our design and documentation accurately reflect the current state of the project and will be a valuable resource for anyone involved in its development.

1.3 VnV Plan and Report

Several updates have been made to the Verification and Validation (VnV) plan and report. Firstly, the traceability matrices in the plan and report have been updated to ensure that all requirements are properly tracked and tested. In addition, discrepancies in the test cases between the VnV plan and report have been corrected. This has been achieved by matching all tests to their corresponding results, ensuring that there are no missing or duplicate tests. Furthermore, reflections that were originally missing in the VnV plan have been added. These reflections cover the skills and knowledge that are expected to be learned during the project verification and validation process. Finally, tests have been given a quantitative

measurement. This will enable the project team to measure the effectiveness of the testing process and to identify any areas where improvements can be made.

2 Design Iteration (LO11)

2.1 Printed Circuit Board Design

Robust electrical connections between the electronic components of the device were a major point of focus for our project. The main electronics requiring robust power and signal electrical connections were the Arduino (microcontroller), ESP8266 (Wi-Fi module), and micro-SD card (local memory storage).

The first iteration in electrical connections occurred between the Proof of Concept and Revision 0 Demonstrations through the transition from a breadboard and jumper wire electrical implementation to a custom design PCB. The power and signal connections made on the breadboard were translated onto an electrical schematic in KiCad, a schematic capture and PCB design software. The PCB layout and trace routes were then created using the schematic to complete the PCB design.

The second iteration in electrical connections occurred between the Revision 0 and Revision 1 Demonstrations through the transition from a custom design PCB to a custom design PCB with a smaller form factor. The second custom PCB design achieved a reduction in board size by 53% from 151 mm x 112 mm down to 97 mm x 81 mm. The use of both planes of the PCB to solder the electronic components was the primary driver in area reduction from Revision 0, which required all electronic components to solder onto the same plane.

2.2 Chassis Design

During the design phase of the device's chassis, the team meticulously considered its size and form factor. The primary objective was to create a sturdy housing that would securely house all of the essential components, including the Arduino, ESP8266, and micro-SD card reader, while shielding them from external elements.

The first version of the chassis was developed between the Proof of Concept and Revision 0 stages. Initially, a pre-made cardboard box was used. However, this was later replaced with a smaller, custom-designed 3D printed chassis with large holes for all the ports and connections.

The second iteration was developed between Revision 0 and Revision 1. The aim was to create a more compact and durable chassis. With the reduction in the size of the PCB, the team was able to decrease the chassis size by 53%. It went from $1.36 * 10^6 mm^3$ to

$0.64 * 10^6 mm^3$. Additionally, the second version featured a more uniform design as it was printed as a single solid piece

2.3   Sensor Configuration Interface

Automating the process of configuring sensors for each test was of utmost importance to make the user experience easy and smooth. Configuring different sensors for each test requires adjusting the Arduino code that is flashed on the board to accommodate the newly mounted sensors. The main objective of the sensor configruation interface was to create a process where users are able to configure the sensors they want to use through Formulate's desktop application instead of manually editing code.

Revision 0 of Formulate expected users to manually ajust the Arduino code that is flashed onto the board every time a different configuration of sensors was required for a new test.

Revision 1 completely changed this process by no longer requiring users to interact with Arduino code to configure sensors. This was achieved by creating a new page in the desktop application called 'Configure Sensors' where users are able to select a couple of parameters such as names of readings, pin number, and units of measurement for the sensors they want to configure. Once users click on 'Generate Arduino Code', the backend of the application would go and edit the Arduino code to reflect the new sensor configuration. This is essentially done by using Python's ability to read and write content to files, which in this case was the Arduino code being flashed onto the board.

2.4   Data Visualization Page

Creating our data visualization page that the user could understand and use from their testing was an essential part of a creating a complete project. The primary objective when creating our data visualization page was to communicate complex data in a clear, concise and engaging way.

Revision 0 of our data visualization page we created a dashboard where the user can go through see all the data in very basic and simple design. It had a some basic kpi metrics and also graphs that helped the user understand what they were looking at.

For the second iteration of our design, between Revision 0 and Revision 1, our goal was to essentially improve the usability of our dashboard. We created a completely new design and layout for Revision 1 allowing for our graphs and KPI metrics to stand out for the user. We also updated our labeling system allowing our dashboard to be more navigable.

3   Design Decisions (LO12)

3.1   Custom Printed Circuit Board

The choice to design a custom PCB to interconnect all electrical components was driven by a PCB's ability to maintain electrical connectivity in high vibration environments. Particularly in McMaster Formula Electric's application as a vehicle driving on top of a road, the device was expected to maintain functionality in high vibration environments such as high vibrations due to rough road surfaces.

Furthermore, PCB's provided a space effective solution to electrical connections relative to jumper wires connected to a breadboard circuit because of the planar copper connections between components. As a result, the overall height required by the electrical circuit was minimized using a 2-layer PCB. The PCB's permanent connections were also desirable for power and signal connections between the Arduino, local memory module, Wi-Fi module, and terminal blocks because those component and their respective connections were always the same irrespective of the test setup.

3.2   3D Printed Chassis

When the team was considering the manufacturing process for the device's chassis, purchasing a preexisting enclosure was initially deemed unsuitable. This decision was driven by the realization that a ready-made chassis would not be able to hold all the essential components securely within the smallest possible form factor, which was a crucial requirement for the device. Instead, the team opted to design and print the chassis themselves, which provided the freedom to tailor it to the specific dimensions of their components.

The custom design allowed for a perfect fit, ensuring that the major components, such as the Arduino, ESP8266, and micro-SD card reader, would be safely housed within the device. Moreover, the team was able to create custom pass-through holes and slots for ports. 3D printing allowed us to create a a very precise and strong chassis with a tolerance of 0.1mm.

Overall, the decision to design and print the chassis in-house enabled the team to overcome the limitations of preexisting enclosures, resulting in a more compact, robust, and customized solution.m.

3.3   Code that Writes Code

During the testing session with McMaster's Forumla E members, they were asked to setup the Formulate device for a test on their own after watching Formulate members perform the setup. Formula E members found configuring sensors to be extremely daunting due to editing the Arduino code that was flashed on the board. The process of editing code not

only requires previous programming experience but also knowledge of Arduino code and how it interfaces with the board. Therefore, it was only fair for the Formula E members to find this task challenging since that knowledge is not a prequisite for them.

After Formula E's feedback, it became clear that there was a need for a method where users are not manually interacting with code everytime a different sensor configuration is setup. The team decided to use the backend code of the application to adjust the Arduino code. This was optimal for a variety of reasons including the removal of manual code editing, shorter setup time, and conveniency for users as they already control the Forumlate device using the deskop application. After the 'Configure Sensors' page was setup, the team also found it much easier to do testing as the configuration of sensors became much more streamlined. In conclusion, implementing the idea of 'code that writes code' was a great succcess in terms of tackling the challenges that came with manual code adjustments for sensor configuration.

3.4   Data Visualization Page

Initially when determining what we should do for our data visualization page our team wanted to create a data analytics website where the user would navigate through their test through a website we created. While creating and testing our data analytics website it became evident that this method would not be suitable for our target user. The website became too complex, hard to navigate and lacked interactivity and adding this would essentially only make it more complex. Our team then decided to use a data visualization tool called Power Bi for our data visualization page. This was done for multplie reasons the mains reasons were that it allowed the user interact with the data more effectively and the usability of a Power Bi dashboard was found to be a lot better than that of data analytics website.

After receiving Formulate E's feedback regarding our data visualization page it was clear that our layout and design was not optimal for what the user wanted. So we decided to change the layout and design using the user feedback to improve its navigability. Our design also followed the same colour scheme and layout of our UI application making it more smooth for the user to go from the UI to the data dashboard. Upon improving the layout and design of our application we were able to identify which KPIs and graphs to keep and how to visualize them better so the user can better interpret the data that they are looking at. Finally we decided to give them the template we created for their dashboard so now it is customizable and the user will be able add other visualization if needed.

In short, through wcreating numerous ways to visualize data our team was able to make decision that we feel is optimal for our user to navigate and interpret their data in an easy and effective way.

4 Economic Considerations (LO23)

Formulate is an open source product for any University's Formula team. Our public GitHub repository allows teams to freely download the Gerber files to manufacture the PCB, chassis CAD model to 3D print the enclosure, executable to run the user interface, and PBIX file for the data visualization template.

The market for Formulate is sizeable, with over 600 Formula series teams (combustion, hybrid, and electric) active across more than 20 countries, with 75 of those teams competing in the Formula Electric competition in the 2023 North American series alone.

Our team plans to create a promotional video on social media platforms such as LinkedIn to attract the attention of the global market of Universities who compete in the Formula Electric competition. Even within the Formula Electric team at McMaster, we aim to attract the attention of members through testing sessions with the product to increase product exposure and educating interested members on the benefit of the product.

The overall capital expenditure (Capex) for teams to build a Formulate device was intentionally small to increase the product's economic viability for all Formula teams. The current cost to purchase the electrical and mechanical hardware using our open source build files is $50 Canadian. The software cost for the user interface and the data visualization platform is $0. Our team estimates the build time with the required hardware at hand to be 2 hours, assuming reasonable workmanship skills at a senior University student level.

5 Reflection on Project Management (LO24)

5.1 How Does Your Project Management Compare to Your Development Plan

Our team followed the weekly Monday team meeting plan consistently throughout the duration of the course. The consistency with which our team followed this plan was a result of the need to share and receive updates on tasks with upcoming deadlines or tasks blocking the progress of sequentially related tasks.

The team communication plan and workflow plan were also followed throughout the course, similarly to the team meeting plan. The team found consistency with the plans because the team had experience with chat messaging through Microsoft Teams and GitHub's issue tracker through course work or internship experience. As a result, our team was comfortable following these plans.

Team member roles evolved naturally throughout the duration of the course to reflect

interests gained as the project progressed. With that said, each member's role was maintained from November 2022 until the end of the course due to the increased aptitude in the sub-system's specific development and increased ability to quickly meet technical goals.

5.2 What Went Well?

Meetings at a weekly frequency was a good process for our team because we were able to find the appropriate balance between having frequent meetings to update the whole team on progress and having adequate development time to provide notable updates during meetings.

In addition to weekly meetings, using GitHub's issue tracker to streamline our workflow helped our team create and assign specific work modules more efficiently. As a result, our team was able to create a systematic process to assign and complete work modules.

All technological tools outlined in the Development Plan were effective for our team, but specific technologies such as Autodesk Inventor, KiCad electrical schematic and PCB editor, PyQT Designer, and Microsoft Power Bi were very effective because of the relative ease of use to achieve functional prototypes and the large amount of readily available documentation for development. This enabled our team to be agile with feature development and overall project iteration.

5.3 What Went Wrong?

The electrical and mechanical hardware prototyping should have begun earlier in the course. Our team began to prototype the device 2.5 months due to time spent creating documentation after the course began, resulting in lost time that could have been used in additional prototype iteration and feature designs.

5.4 What Would you Do Differently Next Time?

Upon reflection, our mechanical and electrical design members should have increased the workload intensity at the start of the course to work on the hardware prototyping in parallel with additional deliverables. This would have allowed the team to achieve more iteration revisions and a more refined final project.