4.0 Software Development Plan

By Leo Dai, Lawrence Benitez, Aiden McDougald, Conner-Ryan Petersen, Evan Steinhoff

4.1 Plan Introduction

The primary goal of this project is to develop software that will control various systems of an electric Formula SAE vehicle, including the active suspension, steering, and telemetry systems. The software will manage car dynamics through a Central Control Unit (CCU) and several subsystem controllers.

The rationale for this project stems from the need for specialized, real-time control software that is customized for Formula SAE vehicles. The system will be similar to the ECUs in consumer electric cars but without unnecessary features, providing precise control and telemetry data crucial for race performance. The project incorporates various software components, such as sensor input handling, data aggregation, and real-time telemetry visualization, ensuring that the driver and the team can monitor the vehicle's performance efficiently.

Activities in the development process include coding and integrating various control units (e.g., active suspension and steering), implementing a telemetry system that collects real-time data during races, and developing a graphical user interface for data visualization. The project also involves extensive testing and debugging to ensure safety and reliability, given the high stakes involved in motorsports.

4.1.1 Project Deliverables

- 4.1.1.1 Database CSC - November 3, 2024
  - This deliverable covers the development of the Database Computer Software Configuration (CSC), responsible for managing and storing telemetry data collected from the vehicle.
- 4.1.1.2 Telemetry GUI CSC - December 8, 2024
  - This deliverable involves the development of the Telemetry Graphical User Interface (GUI), allowing users to view live data and diagnostics from the car in a visually accessible manner.
- 4.1.1.3 Input Observer CSC - January 26, 2025
  - The Input Observer CSC will handle capturing and processing input data from the vehicle's steering and pedals, converting analog signals to digital data for the Central Control Unit (CCU).
- 4.1.1.4 Data Management CSC - January 26, 2025
  - The Data Management CSC focuses on controlling the data flow between various systems within the vehicle, ensuring reliable and timely data transmission.
- 4.1.1.5 Traction Control CSU - March 2, 2025

        - The Traction Control Computer Software Unit (CSU) will be responsible for managing wheel traction, preventing wheel slippage by adjusting power to the drive wheels based on real-time data.
- 4.1.1.6 Virtual Differential CSU - March 2, 2025
    - The Virtual Differential CSU manages power distribution between the vehicle's drive wheels, improving handling during cornering by optimizing power to each wheel.
- 4.1.1.7 Suspension Motor Controller CSC - April 6, 2025
    - This deliverable focuses on the development of the Suspension Motor Controller CSC, which manages the vehicle's active suspension system, allowing for real-time adjustments based on track conditions.
- 4.1.1.8 Steering Wheel Unit Processor CSC - April 6, 2025
    - The Steering Wheel Unit Processor CSC will handle communication between the driver and the Central Control Unit, processing driver inputs and displaying data on the steering wheel interface.
- 4.1.1.9 Safety Check CSC - April 6, 2025
    - The Safety Check CSC ensures that all systems in the vehicle are functioning correctly, with continuous monitoring to detect any faults and trigger necessary safety responses.
- 4.1.1.10 Power Draw Monitor CSC - April 27, 2025
    - The Power Draw Monitor CSC tracks and monitors the power consumption of various systems within the vehicle, ensuring efficient power usage and providing telemetry on energy consumption.
- 4.1.1.11 Graphics Interface CSC - May 4, 2025
    - This deliverable involves the development of the Graphics Interface CSC, responsible for displaying crucial telemetry and system data to the driver in an easily readable format.
- 4.1.1.12 Suspension Kinematics CSU - May 4, 2025
    - The Suspension Kinematics CSU will manage the vehicle’s suspension dynamics, calculating optimal stiffness and damping for the suspension system based on real-time driving conditions.

4.2 Project Resources

4.2.1 Hardware Resources

Resource	*Development*	*Execution*
Windows Computer		
AMD Ryzen 7 7800X3D	X	
64GB DDR5 RAM, NVME SSD, Wifi Connection	X	
RTX 4070 Ti Super 16GB VRAM, 4K Minimum Graphics	X	
Arduino	X	X
Raspberry Pi	X	X
Motor and Motor Controller	X	X
Spring Pack	X	X
Mounting Hardware	X	X

4.2.2 Software Resources

Software
GitHub Version Control
Visual Studio Code IDE
GNU C++ Compiler
Arduino IDE
Linux Operating System

4.3 Project Organization

4.3.1 Project Lead - Aiden McDougald

- The project lead is in charge of making sure the project is on track by assigning tasks based on each team member's skills. The project lead makes sure each team member is working on their assigned tasks on track to completion time. The project lead is also responsible for integrating tasks being worked on by different team members. For example, in the telemetry app, the project lead will be responsible for successful integration between the database and the GUI client.

4.3.2 UI/UX Developer - Lawrence Benitez

- The UI/UX developer is in charge of designing and developing the user interface aspect of the software. In particular, this means designing and developing the front end of the Telemetry App which will allow us to parse through data such as steering angle, tire temperatures, or even GPS data to monitor the car's performance. This role is also in charge of designing and developing the steering wheel unit which shows vital information about the car to the driver.

4.3.3 Software/Backend Developer - Leo Dai

- The backend developer is in charge of developing and maintaining the database used to record telemetry data. The backend developer is also in charge of making sure the database is deployable on a Linux-based system and accessible by other software. This role is also in charge of developing software for critical systems of the car. In particular, this means developing the suspension motor controllers.

4.3.4 Software Developer - Conner Ryan Petersen

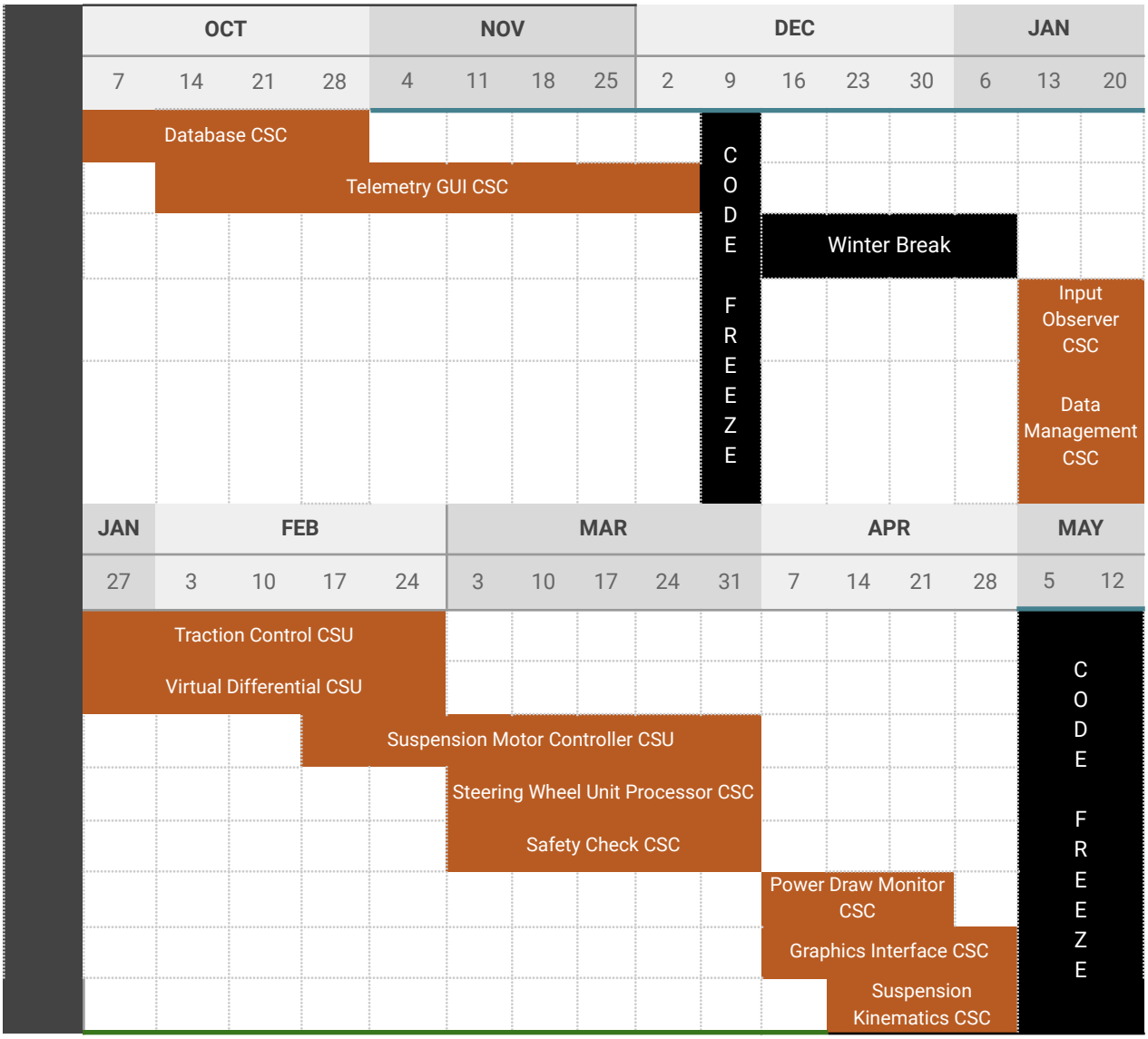
- The software developer is in charge of developing software needed throughout the car. This means the various software components used by the CCU (Central Control Unit) and SPU (Steering/Pedal Unit). This means writing code for Linux-based systems and Arudinos.

4.3.5 Quality Engineer - Evan Steinhoff

- The quality engineer is in charge of making sure the software developed is up to standard with the project lead's demands. This means writing unit tests to test different components of the car to ensure they are robust. The quality engineer in this team is also in charge of writing the safety system in the car.

4.4 Project Schedule

4.4.1 GANTT Chart



4.4.2 Task/Resource Table

Task	People Involved	Hardware Required	Software Required
Database CSC	Leo Dai, Aiden McDougald	Laptop/Desktop running Windows	PostgreSQL, Windows OS

Input Observer CSC	Aiden McDougald, Conner-Ryan Petersen	Raspberry Pi, Steering and Pedal Sensors	C++, Arduino IDE
Data Management CSC	Evan Steinhoff, Aiden McDougald	Raspberry Pi, Sensors	C++, Data Management Libraries
Telemetry GUI CSC	Lawrence Benitez, Evan Steinhoff	Laptop/Desktop with Display	Go, Fyne.io Framework, Windows OS
Traction Control CSU	Aiden McDougald, Conner-Ryan Petersen	Raspberry Pi, Motors, Traction Sensors	C++, Traction Control Algorithms
Virtual Differential CSU	Evan Steinhoff, Aiden McDougald	Raspberry Pi, Differential Sensor, Motor Controllers	C++, Control Software
Suspension Motor Controller CSC	Leo Dai, Aiden McDougald	Raspberry Pi, Brushless DC Motors, Suspension System	C++, Motor Control Libraries
Steering Wheel Unit Processor CSC	Lawrence Benitez, Evan Steinhoff	Raspberry Pi, LCD Display, Steering Wheel	C++, Linux, Steering Control Software
Safety Check CSC	Evan Steinhoff, Conner-Ryan Petersen	Raspberry Pi, Safety Switches, Sensors	C++, Safety Monitoring Software
Power Draw Monitor CSC	Evan Steinhoff, Leo Dai	Raspberry Pi, Power Sensors	Python, Power Monitoring Libraries
Graphics Interface CSC	Lawrence Benitez, Conner-Ryan Petersen	Raspberry Pi, LCD Display	Python, GUI Framework (e.g., Tkinter)
Suspension Kinematics CSU	Aiden McDougald, Leo Dai	Raspberry Pi, Brushless Motors, Sensors	C++, Kinematics Algorithms