Statement of Work (SOW)

SWE Grand Prix - CS 321 - Spring 2023

R. Pettit

1 General Information

1.1 Scope of Work

The SWE Grand Prix project will develop the software for a WiFi-controlled race car competition. Five (5) teams will develop software to control race cars with identical hardware. One (1) team will develop the race management software that acts as the central hub for race communication, status, and scoring. This race management team will consist of one representative from each of the five racer teams.

The development effort will proceed using the Scrum approach to agile software development. Dr. Pettit will act as your customer.

1.2 Background

The learning objective for the CS321 term project is to gain experience working in a realistic team environment to develop a software product. Unlike the relatively simple programs you may have worked with up to this point, the CS 321 term project will:

- Employ an organized development method (Scrum)
- Apply a team-based approach to development
- Require the use of configuration management and project planning tools
- Require the integration and adaptation of legacy software
- Require communication and integration across teams

Additionally, the CS 321 term project will take many of you out of your comfort zone as you work with embedded systems rather than desktop or web-based applications.

2 Team Requirements

2.1 Race Cars

The top-level requirements for the race cars are identified below. You will need to expand on these to identify high level software *features* and then specific *user stories* to implement the features.

- Each race car shall be controlled remotely via WiFi
- The remote control software shall be implemented on a laptop computer and use a graphical user interface (GUI)
  - Alternatively, the remote control software may be implemented with a joystick-style controller attached to the laptop computer
- The onboard computer for each race car shall consist of a BeagleBone Black single-board computer, hereafter referred to as the SBC
  - Live video shall be captured via a USB webcam attached to the SBC
  - Video shall be streamed to Race Management during the race
  - Steering shall be controlled via pulse width modulation (PWM) signals from the SBC to the steering servo
  - The steering servo shall be connected to pin P9_14 on the SBC
  - Driving (forward and reverse) shall be controlled via PWM signals from the SBC to an Electronic Speed Controller (ESC), which then controls the drive motor
  - The ESC shall be connected to pin P9_16 on the SBC
  - Control for steering and driving operations shall allow for incremental adjustments (i.e. variable speed and steering angles)
- At the beginning of a race, each car shall report to Race Management to indicate they are ready to begin the race
- Race cars shall await a signal from Race Management before crossing the start line of a race
- Each car shall stay within the boundary of the course
- Cars shall not collide with other race cars
- Cars shall race for a predetermined number of laps

2.2 Race Management

The top-level requirements for the race cars are identified as follows.

- The Race Management (RM) team shall consist of one member from each of the Racer teams

  - Each RM team member shall be permanently assigned to RM for the duration of the project
  - RM team members are responsible for coordinating with their respective Racer teams to define and document the interfaces between race cars and race management.

- RM shall provide a central interface to manage a race
  - The RM interface shall accommodate live video feed as well as live telemetry from any additional sensors that may be added to the Racers
  - The RM interface shall provide a mechanism for race cars to check in for the start of a race
  - The RM interface shall provide a start signal to enable race cars to begin racing

- Race Management shall provide a graphical user interface to display live race results using the classroom projector or large monitor via an HDMI connection

- Race Management shall allow statistics to be displayed in English or Metric units, configurable at the start of a race

- Race Management software shall be portable across Windows, Mac, or Linux operating systems

- Race Management shall score the race according to completion time
  - Race Management shall institute a 1-second penalty for each time a car travels outside the racetrack boundary
  - Race Management shall institute a 2-second penalty for a roll-over event
  - Race Management shall institute a 0.5 second penalty for collisions
    * This penalty shall be assessed against the trailing car in a collision.

- Race Management shall manage the overall competition using a tournament-style bracket in which racers compete two at a time

3 Customer Furnished

Each race team will be provided with a 1/12 scale race car with steering servo and ESC connected to the onboard computer. The computing platform will be a BeagleBone Black with USB connections for WiFi and a 1080P web camera. Additional sensors may be added later.

4 Deliverables

Deliverable	Frequency	Description
Sprint Reflections	Every 2 weeks at the end of each sprint	For the sprint you just finished, briefly document (using bullets) • Who was team lead / Scrum Master for this sprint? • What went well? • What went poorly? • What do you hope to do differently for the next sprint? • What was your predicted velocity? • What was your actual velocity? • List of user stories completed this sprint • Include a screen shot of your Scrum board • Provide an overall summary of your defined features and user stories • Code metrics (starting with Sprint 2)
Design Review	Once at ∼Week 10	Each team gives a 15-min overview of their architecture, design, and current status Submit PDF slides with: • Drawing of your top-level architecture • UML class diagram • At least one sequence diagram associated with a user story • One state machine capturing race operations from the perspective of a race car or race management • Race Management should provide an overview of the interface design

Final Project Documentation	Once - Last week of classes	Each team provide a summary of their project including: • Summary of lessons learned • Planned vs. actual accomplishments • As-built design documentation • Code with instructions on how to install and run • Testing report describing how you tested your software, the code coverage of your tests, and open vs. closed defects
Project Demonstration	Once - During Final Exam Block	Tournament style race, 2 cars at a time

5 Contractor's Key Personnel

Each racing team will consist of 7 students. The Race Management team will then consist of 5 students, with one representative from each racing team. One student per sprint will serve as the Scrum Master so that each team member has at least one turn in this role through the semester. All team members are responsible for software development. Each team member is required to provide via Blackboard. This should be submitted as Writing Assignment 1.

6 Security Requirements

In this class, it is *perfectly acceptable and expected* to use open source software or libraries or even use code that you find on sites such as Stack Exchange, etc. This is what you would be doing in the "real world" and I do not consider this cheating at all. *However*, for any 3rd party software you use, you must cite that software in your source code comments and you must appropriately test that this software will work under the desired use cases of your project.

Each team should take measure to make their software as secure as possible to avoid any hacking attempts.

Additionally, for group project assignments, you are **NOT ALLOWED** to include "guest names." Every person listed as a collaborator must contribute. If someone is listed as a collaborator but did not contribute, all will be given a zero on the assignment and reported to the university honor committee.

7 Data Rights

The customer shall retain all data rights to submitted projects.