SOFTWARE REQUIREMENTS SPECIFICATION

for

Train Control System

Version 1.1

Prepared by: Team 8

Andrew White **Sara Keriakes**
Grace Keywork **Tim Bottegal**
Jay Fu **Johann Traum**

2025-09-27

Contents

1 Introduction

1.1 Purpose

This document will outline all software requirements for the Train Control System (TCS). It will provide an overview of the final deliverable and will specify both functional and non-functional requirements.

This document will also serve as the contributors' main tool of clarifying project requirements with users and stakeholders.

1.2 Scope

This Software Requirements Specification (SRS) defines the scope of the Train Control System (TCS), a system designed to manage and control train operations across railway networks. The system is comprised of several modules: The Centralized Track Control (CTC) Office, The Track Controller (Wayside Controller), Track Model, Train Model, and Train Controller.

The primary goal of the TCS is to ensure passenger safety while facilitating the efficient and reliable movement of trains while conforming to the needs and expectations of our users and other stakeholders. Key users of the TCS include the Programmer, Dispatcher, Driver, Passenger, Murphy, Track Builder, and Engineer.

1.3 Definitions, Acronyms, & Abbreviations

1. **SRS**: Software Requirements Specification
2. **TCS**: Train Control System
3. **PLC**: Programmable Logic Controller
4. **RFID**: Radio Frequency Identification
5. **UI**: User Interface

6. $\mathbf{K_i}$**:** Integral Gain
7. $\mathbf{K_p}$**:** Proportional Gain
8. **Vital**: Processes that are critical for maintaining safe operation of the Train Control System
9. **CTC Office**: Centralized Traffic Control Office
10. **Wayside Controller**: Track Controller
11. **Contributors**: Team 8 members responsible for the design and implementation of the Train Control System.
12. **Murphy**: User whose main goal is to cause harm to the Train Control System

1.4 References

There are no references to outside material at this time.

1.5 Overview

This SRS is divided into two sections: Overall Description and Specific Requirements. The Overall Description section provides a high-level overview of the TCS, outlining its functions. This section also discusses the users of the system, outside constraints to the design of the system, and the Contributors' ethos on the system's design. The Specific Requirements discusses the system's functions and requirements at a low level, both functional and non-functional in nature. It lists the granular requirements of each module in easily testable statements.

2 Overall Description

2.1 Product Perspective

This system is a self-contained train simulation environment. The key modules of the simulation environment include a CTC Office interface, Track Controller software interface, Track Controller hardware interface, a Track Model, a Train Model, a Train Controller software interface, and a Train Controller hardware interface. These modules will then be integrated and work together to perform the overall simulation.

2.2 Product Functions

The system will be able to safely and automatically guide trains along transit lines from an uploaded schedule, as well as accept manual input from users.

Per-Module Features

- **CTC Office:** Control scheduling and dispatching trains, set suggested speed and authority for all trains in the system, and display the state of each block on the entire rail system.
- **Track Controller:** Receive information from the CTC Office, including route and switch plans, maintenance closures, information on suggested speed and authority. Issue field device commands to the Track Model, such as setting switch alignments, signal aspect, and crossing gate commands, while simultaneously receiving track status and feedback from the Track Model.
- **Track Model:** Maintain a flexible model of the railway layout incorporating blocks, signals, switches, and stations.
- **Train Model:** Simulate the physical dynamics of the train on the track, including velocity, acceleration, and mass.

- **Train Controller:** Regulates train velocity according to issued commands without surpassing authorized limits. Also oversees door operations and internal temperature control.

2.3 User Characteristics and Expectations

Programmer

Name: Alex Hall
Age: 63 years old
Biography: Born to a wealthier household in Collier Township. Previously lived in Minnesota, and then moved to California for 4 years. Has a wife and kids.
Education: PhD in Mechanical Engineering from Carnegie Mellon University
Salary Range: $120,000 yearly
Residence: Homeowner in Shadyside, Pittsburgh
Technology Comfort Level: Prefers older technology, takes time to adjust to modern systems.
Hobbies: Going to Disney World, golf, football enthusiast, sports gambling.
Goals: Ensuring the track is functional and his program works.

Dispatcher

Name: Thomas White
Age: 40 years old
Biography: Thomas was in Ohio and lived there throughout high school. He was the top student in his class and took more STEM-heavy classes. He is religious and attends Synagogue. He is married with children.
Education: Bachelor's Degree in Computer Science, University of Pittsburgh
Salary Range: $80,000 yearly
Residence: Homeowner in Shadyside, Pittsburgh
Technology Comfort Level: Comfortable with modern technology.
Hobbies: Walking, hiking, exploring Pittsburgh, carpentry
Goals: Dispatching a train, and ensuring it makes it safely to its destination.

Driver

Name: Jeffery Smith
Age: 56 years old
Biography: Born and raised in the Southside of Pittsburgh, where he has lived his entire life.
Education: High School Diploma / GED
Salary Range: $60,000-$70,000 yearly
Residence: Renter in Southside, Pittsburgh
Technology Comfort Level: Slow to adopt new technology and will only use his computer when necessary, such as for online purchases. He prefers more traditional media, like listening to the radio.
Hobbies: Jeffery enjoys making things with his hands, taking care of his lawn, and supporting his kids at their events.
Goals: To drive as safely as possible and cause no harm.

Passenger

Name: Harper Jones
Age: 21 years old
Biography: Harper grew up in the South Hills of Pittsburgh and have lived there all her life. She is a paraplegic and uses a wheelchair.
Education: Bachelor's Degree in English, University of Pittsburgh (In progress)
Salary Range: None
Residence: Lives with parents in South Hills, Pittsburgh.
Technology Comfort Level: High. Commonly uses a smartphone and applications on the internet. Loves using social media and is an avid poster.
Hobbies: Drawing, reading historical fiction, hanging out with friends.
Goals: To have step free boarding, reliable elevators/ramps, ample space for wheelchairs, and minimal gaps between the train station and train door.

Murphy

Name: Henry Murphy
Age: 31 years old

Biography: Born in Summit, NJ to a wealthier household. Was an average student at Summit High School but was an athletic three season athlete. Began practicing malicious magic at age 24.
Education: Bachelor's Degree in Business, University of Pittsburgh
Salary Range: $90,000-$110,000
Residence: Renter in Downtown, Pittsburgh
Technology Comfort Level: High. Commonly uses a smartphone and applications on the internet.
Hobbies: Dark incantations, watercolor painting, DnD
Goals: After drudging through his day as an analyst, he cast spells on the trains to cause chaos for passengers during rush hour to maximize misery for others.

Track Builder

Name: Robert Wilson
Age: 42 years old
Biography: Born and raised in Congers, NY. He always enjoyed math and history during high school, though chose not to pursue either in college. He later moved to Boulder, CO for his studies, then to Cheslea, NY to work for the MTA. He eventually settled in Friendship, PA with his wife.
Education: Bachelor's Degree in Civil Engineering, University of Colorado Boulder
Salary Range: $60,000-$90,000 yearly
Residence: Renter in Friendship, PA
Technology Comfort Level: Very comfortable. He is very into building personal computers due to his love of gaming.
Hobbies: Playing Counter Strike, baking.
Goals: Ensure all tracks are properly and safely installed. Build all tracks to adhere to desired specifications. Avoid train delays by completing work efficiently and on time.

Train Engineer

Name: Liam Jacobs
Age: 36 years old
Biography: Born and raised in Clay County, West Virginia to a lower income family.

Has always been interested in working with his hands and machinery. He later bought a home in Bloomfield, PA with his wife and 4 dogs.
Education: Bachelor's Degree in Electrical Engineering, University of Pittsburgh
Salary Range: $90,000-$120,000 yearly
Residence: Homeowner in Bloomfield, PA
Technology Comfort Level: Very comfortable. Uses his phone for most tasks.
Hobbies: Weightlifting, cooking, maintaining fish tanks.
Goals: Set the proper K_P and K_I for the train within the onboard control system.

2.4 Constraints

1. The system shall run on Windows 10.
2. All modules shall be able to run at the same time.
3. All modules shall work together to not conflict with actions of another module.
4. All modules shall be written using Python 3.12+.
5. All modules shall react safely to all failure modes.
6. All actions shall reflect the state of the system.
7. All modules shall use realistic units to simulate realistic train.

2.5 Assumptions and Dependencies

Murphy's Law

There will be issues with the system that the Contributors will not be able to predict, whether structural or technical in nature. This will require changes to our implementation and therefore documentation.

Emergence

As the consultation process proceeds with the Customer and Stakeholders, the Customer may alter their requirements to better suit their needs. The Contributors may also need to alter the system to suit a particular need of a Stakeholder. Misunderstandings may also be uncovered as the consultation process proceeds that will require the Contributors to revise structure or implementation.

Issues in Modular Integration

As the implementation of module integration proceeds, there may be unseen issues with their ability to do so. This may require other internal dependencies not yet known achieve integration.

2.6 Apportioning of Requirements

Changes to Train Parameters

In the scenario that any train parameter requires alteration, the implementation should be generalized to accept a range of parameters.

Changes to Track Layout

In the unlikely scenario that the current track layout is altered before the TCS is completed, the track layout implementation should be easily adjustable and shall not be hard-coded into the current layout.

Changes to User Input

The Contributors have created very specific layouts on how users will interact with the system. If design requirements forces the Contributors to alter the method of user interaction, this will require changes to the structure of the TCS.

3 System Architecture

3.1 External Interfaces

CTC Office

- **Inputs:**
  1. From the Track Controller: Status of Switch positions (bool), Signal states (bool), Block occupancy (bool), Broken rail detection (bool), Crossing status (bool)
- **Outputs:**
  1. To the Track Controller: Suggested speed (m/s), Suggested Authority (m)
  2. To the Dispatcher: Train Status (string), Schedule (string)

Track Controller

- **Inputs:**
  1. From the Programmer: PLC programs (.py files)
  2. From the CTC Office: Suggested authority (m) and Suggested speed (m/s)
  3. From the Track Model: Switch positions (bool), Signal states (bool), Block occupancy (bool), Broken rail detection (bool), Crossing gate status (bool)
- **Outputs:**
  1. To the CTC Office: Track occupancy (bool), switch states (bool), signal aspects (int), crossing gate states (bool)
  2. To the Track Model: Commands for Switch commands, crossing gate commands, signal commands (various python functions)
  3. To the Train Model: Commanded speed (m/s) and Commanded Authority (m)

Track Model

- **Inputs:**
  1. From the Track Controller: Switch commands, crossing gate commands, signal commands (various python functions)
  2. From the Train Model: Block occupancy (bool)
- **Outputs:**
  1. To the Track Controller: Status of switch positions (bool), Signal states (bool), Block occupancy (bool), Broken rail detection (bool), Crossing status (bool)
  2. To the Train Model: Percent Grade (%), Beacon information (string), Number of persons boarding at a station (int)

Train Model

- **Inputs:**
  1. From the Train Controller: Power (kW), service brake command (bool), emergency brake command (bool), door opening/closing (bool), Announcements (string)
  2. From the Track Model: Percent Grade (%), Beacon information (string), Number of persons boarding at a station (int)
  3. From Train Model: Train Mass (kg)
  4. From the Track Controller: Switch commands, crossing gate commands, signal commands (various python functions)
- **Outputs:**
  1. To the Train Controller: Emergency Brake Command (bool), Commanded speed (m/s), Commanded Authority (m), Beacon Information (string)
  2. To the Track Model: Block occupancy (bool)

Train Controller

- **Inputs:**

1. From the Train Model: Emergency Brake command (bool), Commanded speed (m/s), Commanded Authority (m), Beacon Information (string)

- **Outputs:**
  1. To the Train Model: Power (kW), Service Brake command (bool), Emergency Brake command (bool), Door open/close (bool), Announcements (string)

3.2 Functional Requirements

CTC Office

1. The CTC Office shall operate autonomously based on an uploaded schedule.
   (a) The autonomous control of the CTC Office shall be able to be paused at any time by the Dispatcher.
   (b) Currently dispatched trains shall be given the correct suggested authority and suggested speed based on its route.
2. The Dispatcher shall be able to manually dispatch a train one at a time, and manually dispatched trains shall be added to the existing schedule.
3. The Dispatcher shall be able to manually override a train's suggested speed and suggested authority.
4. The Dispatcher shall be able to close track blocks for maintenance.
5. The Dispatcher shall be able to manually override switch positions.
6. The CTC Office shall calculate and display throughput metrics for each line in the system.
7. The CTC Office shall display the current state of the entire transit system for the Dispatcher.

Track Controller

1. The Track Controller shall accept an input of a PLC program written by the Programmer. It shall interpret this file and automatically move switches based on program execution.
   (a) The PLC language shall be based only on Boolean variables.

2. The Track Controller shall detect the presence of a train on a specific block through the use of track circuits.

3. The Track Controller shall detect broken rail segments.

4. The Track Controller shall relay suggested speed and suggested authority from the CTC Office to the Track Model for transmission to the Train Model.

5. The Track Controller shall control the switching of the track.

6. The Track Controller shall control railway crossings.

7. The Track Controller shall report the state of the track blocks, railway crossings, and block occupancy to the CTC Office for display to the Dispatcher.

8. **Dangerous conditions that will cause safety issues (i.e. collisions, overlapping authorities) shall not be allowed to occur, even if caused by a manual override.**

Track Model

1. The Track Model shall load a model (representation) of the track layout.

2. The Track Model shall keep a log of ticket sales.

3. The Track Model shall keep a log of the number of passengers entering and leaving the station.

4. The Track Model shall display block occupancy, switch positions, and light states to the Track Builder.

5. The Track Model representation shall have an environmental temperature set by Murphy.

6. The Track Model shall have track heaters that activate below a certain environmental temperature.

7. The Track Model shall have the following failure states that are activated by Murphy: Broken Rail, Track Circuit Failure, and Power Failure.

Train Model

1. The Train Model, given a power command from the Train Controller, shall calculate a velocity for the train following Newton's laws.
2. The Train Model shall display the following properties to the Passenger: Length, Height, Width, Mass, Acceleration, Velocity, Crew Count, and Passenger Count.
3. The Train Model shall receive a signal through the track circuit with suggested speed and authority and relay it to the Train Controller.
4. The Train Model shall be able to receive beacon (RFID) inputs and relay them to the Train Controller.
5. The Train Model shall regulate its internal temperature given a set temperature from the Train Controller.
6. The Train Model shall have lights and doors operated by the Train Controller.
7. The Train Model shall have an Emergency Brake that is activated by the Passenger or accept an input from the Driver via the Train Controller.
8. The Train Model shall have a Service Brake that is activated by the Driver via the Train Controller.
9. The Train Model shall have an announcement system controlled by the Train Controller.
10. The Train Model shall have the following failure modes that are activated by Murphy: Train Engine Failure, Signal Pickup Failure, Brake Failure.

Train Controller

1. The Train Controller shall regulate the velocity setpoint from either the CTC or Train Driver, depending on the control mode via calculating the power required. It shall then relay this to the Train Model for execution.
2. The Train Controller shall set the internal temperature of the train and send the value to the Train Model for regulation.
3. The Driver shall be able to activate either the Emergency Brake or Service Brake by sending a command to the Train Model.

4. The Engineer shall be able to set and modify the regulation system's K_P and K_I values.

5. The Train Controller shall be able to control the train's lights and doors via commands to the Train Model.

6. The Train Controller shall stop the train at a designated station and relay a message to the Train Model's announcement system.

7. The Train Controller shall not allow the Driver or Engineer to place the train in a hazardous condition.

   (a) The Train Controller shall not allow the Driver to exceed the suggested speed set by the CTC Office.

   (b) The Train Controller shall not allow the Driver to exceed the given authority set by the CTC Office. The Train Controller shall automatically execute a braking maneuver before the given authority is exceed.

   (c) The Train Controller shall not allow the Engineer to set K_P or K_I values that place the train in a hazardous condition.

   (d) The Train Controller shall set the output power to 0 in the case of an emergency brake event.

3.3 Non-Functional Requirements

3.3.1 Performance

CTC Office

1. The CTC Office shall have one user terminal.

2. The CTC Office shall only support a single Dispatcher.

3. The CTC Office shall be able to load and run a single schedule at a time.

4. The CTC Office shall be able to load and process all block occupancy data for the entire system.

5. The CTC Office shall be able to load and process all throughput data from the entire system.

Track Controller

1. The Track Controller shall have one user terminal.
2. The Track Controller shall only support a single Programmer.
3. The Track Controller shall receive suggested speed and suggested authority from the CTC Office for all trains and relay it to the Track Model for further transmission.

Track Model

1. The Track Model shall only have one terminal.
2. The Track Model shall support two users: Track Builder and Murphy.
3. The Track Model shall take the commanded speed and authority from the Track Controller and transmit them to the Train Model via track circuit.

Train Model

1. The Train Model shall only have one terminal.
2. The Train Model shall support two users: Passenger and Murphy.
3. The Train Model shall receive the commanded speed and authority from the Track Model via track circuit and relay them to the Train Controller.
4. The Train Model shall take in a power command from the Train Controller and calculate the train's velocity using Newtonian equations. The Train Model shall then relay this value to the Train Controller.
5. The Train Model shall execute all internal calculations in SI units.

Train Controller

1. The Train Controller shall have one terminal.
2. The Train Controller shall support two users: Train Engineer and Driver.
3. The Train Controller shall receive commanded speed and authority from the Train Model and use them as input.
4. The Train Controller shall use these inputs as well as the actual speed reported by the Train Model to calculate a power command, and relay this to the Train Model.

3.3.2 Reliability

CTC Office

1. The CTC Office shall validate all manual inputs by the Dispatcher.
2. The CTC Office shall validate all schedules uploaded by the Dispatcher.
3. The CTC Office shall periodically check routes, switch alignments, and authority for validity.

Track Controller

1. The Track Controller shall only accept PLC files using exclusively boolean values.
2. The Track Controller shall validate PLC files before program execution.
3. The Track Controller shall always accept user input for manual operation, with the exception of inputs that will cause unsafe conditions.

Track Model

1. The system shall immediately report detected failures to the necessary module(s).
2. The Track Model shall periodically compare heater status to environmental temperature to ensure proper heater function.
3. The Track Model shall accurately display all track properties with correct units and consistency.

Train Model

1. The system shall immediately report detected failures to the necessary module(s).
2. The Train Model shall ensure it accurately reflects real-world dynamics through Newtonian equations.

Train Controller

1. The Train Controller shall periodically compare the power command to the commanded speed to ensure proper control function.

2. The Train Controller shall periodically compare the status of lights, doors, and internal temperatures to commanded values to ensure proper system function.

3. The Train Controller shall always accept manual inputs from the Driver when in manual operation mode, with the exception of inputs that exceed commanded speed or authority.

3.3.3 Availability

CTC Office

1. The current state of all systems (ie. occupied blocks, switch positions etc.) shall be displayed to the dispatcher at all times.

2. In the case of a software failure, the CTC Office shall restart without prompt to the last valid state.

Track Controller

1. The Track Controller shall allow the Programmer to download the current PLC file for reference.

2. In the case of a software failure, the Track Controller shall restart without prompt to the last valid state.

Track Model

1. The Track Model shall always make the track status parameters available to the Track Builder.

2. In the case of a software failure, the Track Model shall restart without prompt to the last valid state.

Train Model

1. The Train Model shall always make the train properties available to the Passenger.

2. In the case of a software failure, the Train Model shall restart without prompt to the last valid state.

Train Controller

1. The Train Controller shall always be capable of carrying out the Emergency Brake function.
2. The Train Controller shall always make the train properties available to the Driver and Train Engineer.
3. In the case of a software failure, the Train Controller shall restart without prompt to the last valid state.

3.3.4 Security

1. Information shall only be passed between modules if necessary for TCS function.
2. Access to items such as GitHub repository, GitHub project, Jira, and OneDrive shall be limited to verified contributors as well as the Professor and TAs.

3.3.5 Safety

CTC Office

1. The CTC Office shall only give authority to any given track segment to a single train at a given time.
2. The CTC Office shall only dispatch one train at a time.

Track Controller

1. The Track Controller shall have the railway crossing lights and gate closed when its track circuit is closed.
2. The Track Controller shall display only one color at a time per block through the traffic signals.
3. The Track Controller shall be able to lower its railway crossing gates in the case of a power or track circuit failure.
4. The Track Controller software and data shall be backed up to non-volatile memory.

Track Model

1. The Track Model shall maintain safe track conditions during adverse weather via the use of track heaters.
2. The Track Model shall detect the following failure conditions: Broken Rail, Track Circuit Failure, and Power Failure.

Train Model

1. The Train Model shall prevent its doors from opening while the train is moving.
2. The Train Model shall detect the following failure conditions: Engine Failure, Signal Pickup Failure, and Brake Failure. These failure detections shall then be relayed to the Train Controller for action.
3. The Train Model shall have an Emergency Brake operable by passengers in case of emergency.

Train Controller

1. The Driver shall be able to activate either the Emergency Brake or Service Brake by sending a command to the Train Model.
2. The Train Controller shall not allow the Driver or Engineer to place the train in a hazardous condition.
   (a) The Train Controller shall not allow the Driver to exceed the suggested speed set by the CTC Office.
   (b) The Train Controller shall not allow the Driver to exceed the given authority set by the CTC Office. The Train Controller shall automatically execute a braking maneuver before the given authority is exceed.
   (c) The Train Controller shall not allow the Engineer to set K_P or K_I values that place the train in a hazardous condition.
3. The Train Controller shall set the output power to 0 in the case of an emergency brake event.
4. The Train Controller software and data shall be backed up to non-volatile memory.

3.3.6 Maintainability

1. The system shall be divided into modules.
2. The failure of one module shall not affect the functionality of other modules, with the exception of pass through data.
3. Contributors shall attend regular scrum meetings to maintain system cohesiveness.
4. Contributors shall follow the set Coding Standard, and proper GitHub commit and push practices shall be followed.

3.3.7 Portability

1. The entirety of the system shall run on Windows 10 from a single executable.

3.4 Design Constraints

1. The system simulation shall run on a 2016 Microsoft Surface Pro running Windows 10 22H2.
2. Any module UI shall not be full screen to accommodate other module UI.

3.5 Additional Requirements

1. The system shall be capable of running at least 10 times faster than wall clock time.
2. The system shall be able to be paused.