Wonderfresh

PAAC North Shore Extension

System Design Document

Version 1.0

March 16, 2017

Rail Control Systems

Prepared for

COE1186 - Software Engineering

Instructor: Joseph A. Profeta III, Ph.D.

Spring 2017

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Description** | **Author** | **Comments** |
| 3/15/17 | Version 1.0 | Rail Control Systems | First version of the System Design Document |
|  |  |  |  |
|  |  |  |  |

**Document Approval**

This System Design document has been reviewed, accepted, and approved by the following:

|  |  |  |  |
| --- | --- | --- | --- |
| **Signature** | **Printed Name** | **Title** | **Date** |
|  | Austin Bagnato | Train Controller |  |
|  | Kevin Carr | Track Model |  |
|  | Sarah Higbee | CTC Office |  |
|  | Angela Hoeltje | Moving Block Overlay |  |
|  | Camille Holland | Train Model |  |
|  | Kayla Walker | Communication Controller, Track Circuit and Train Antenna |  |

**Table of Contents**

**Introduction**

**Purpose:**

This document will give an overview of the architecture and design of the Wonderfresh PAAC North Shore Extension Train System. The entire system will be covered followed by the structural and behavioral details at the subsystem level.

**Scope:**

This documents shall cover the entire Wonderfresh PAAC North Shore Extension Train System and the details of the six modules within: CTC, Communication, Track Model, Train Model, Train Controller, and Moving Block Overlay.

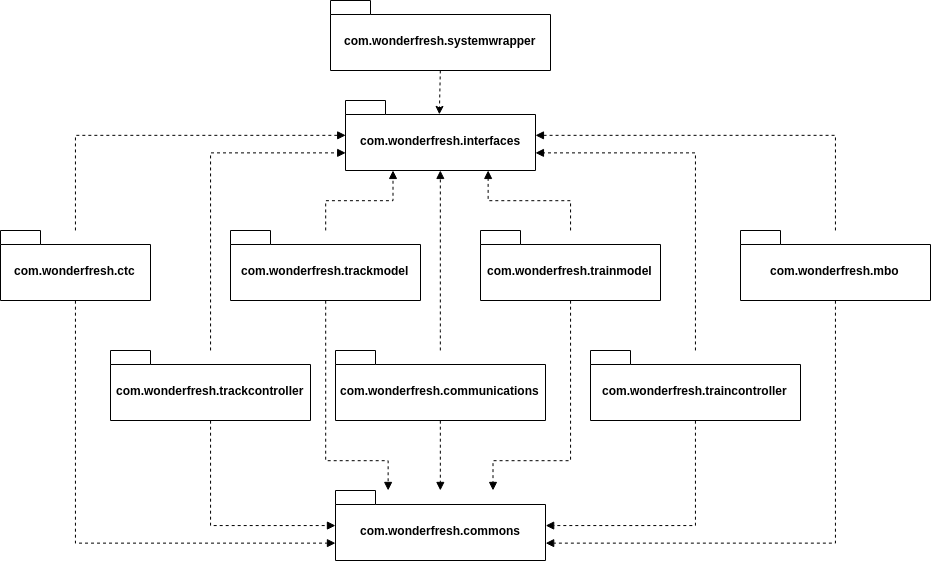
**Context:**

This document represents the architecture of the final system and shall be used to implement the system. Due to a group member recently leaving an unfinished module our design will need to be modified to exclude the track controller module.

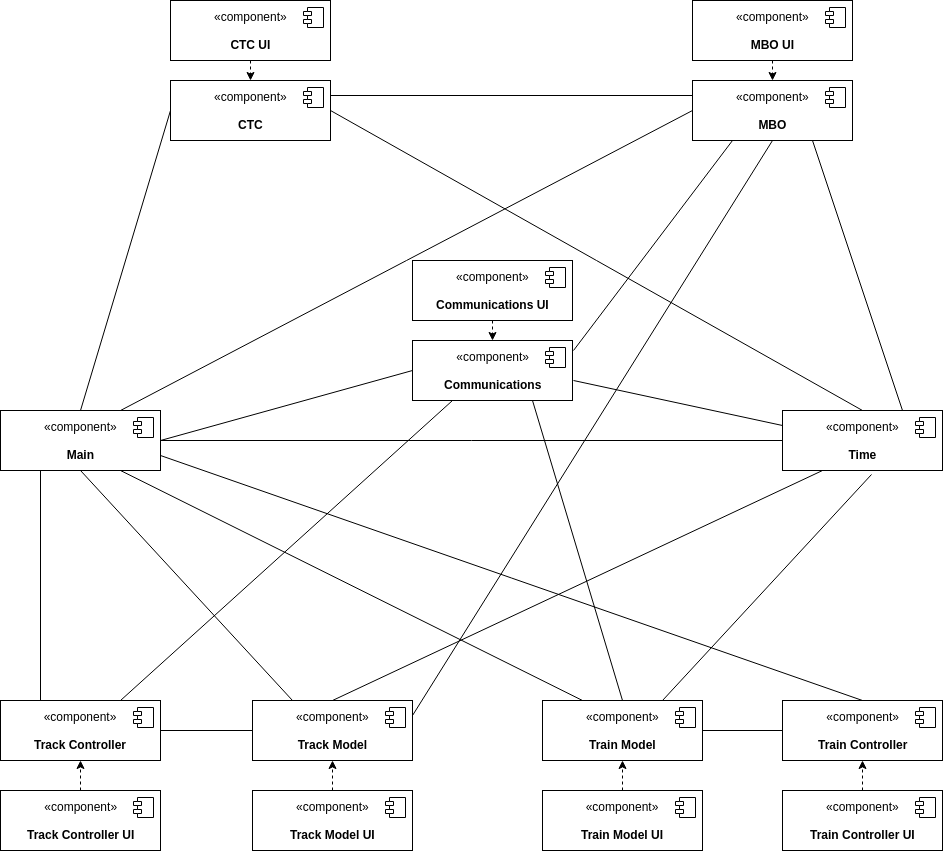
**Summary:**

This document is composed of a package diagram, component diagram, use case diagrams, class diagrams, sequence diagrams, and UI designs. The package diagram and component diagram show the overall structure of the system. There will be use case diagrams each of the modules. There will be a class diagram for the entire system followed by more detailed class models for each module. Each module will also have sequence diagrams and a UI design.

**Package Diagram:**

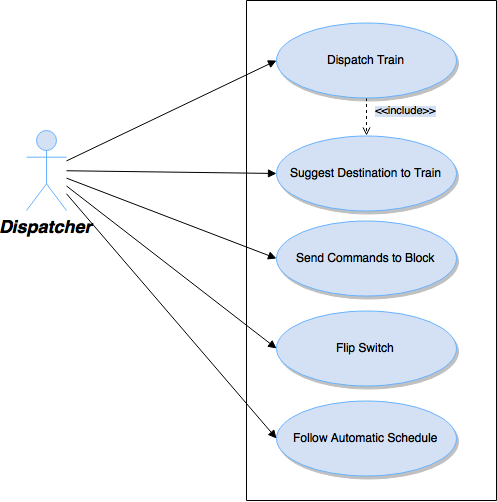


**Overall Component Diagram:**



**Use Case Diagrams**

**CTC:**



***Dispatch Train***

1. User enters a name in the dispatch train name field.
2. User hits the Dispatch button.
3. A train with the given name appears on the map at the starting point in the yard.
4. A train with the given name appears on the train list.

**Extensions**:

3a: A train is at rest on the starting point.

.1: A message displays, saying that the starting point is blocked and that the user can try again when it's free.

.2: User **suggests a destination to the train.**

.3: User may try to dispatch train again.

3b: A train is moving through the starting point.

.1: A message displays, saying that the starting point is blocked and that the user can try again when it's free.

.2: User waits for the train to leave the starting point.

.3: User may try to dispatch train again.

***Suggest Destination to Train***

1. User selects a train from the train list.
2. The train viewer displays the train's current position, suggested destination, and suggested average speed.
3. User edits destination and/or speed.
4. User hits the Send button.
5. The train viewer and map display the train's movement.

**Extensions**:

4a: User entered invalid values for destination/speed.

.1: An error message displays.

.2: User may reenter values.

***Send Commands to Block***

1. User selects a block from the block list.
2. The block viewer displays the block's current state.
3. User edits speed, authority, and/or open/closed status for that block.
4. User hits the Send button.
5. The block viewer displays the block's new state.

**Extensions**:

4a: User entered invalid values for speed/authority.

.1: An error message displays.

.2: User may reenter values.

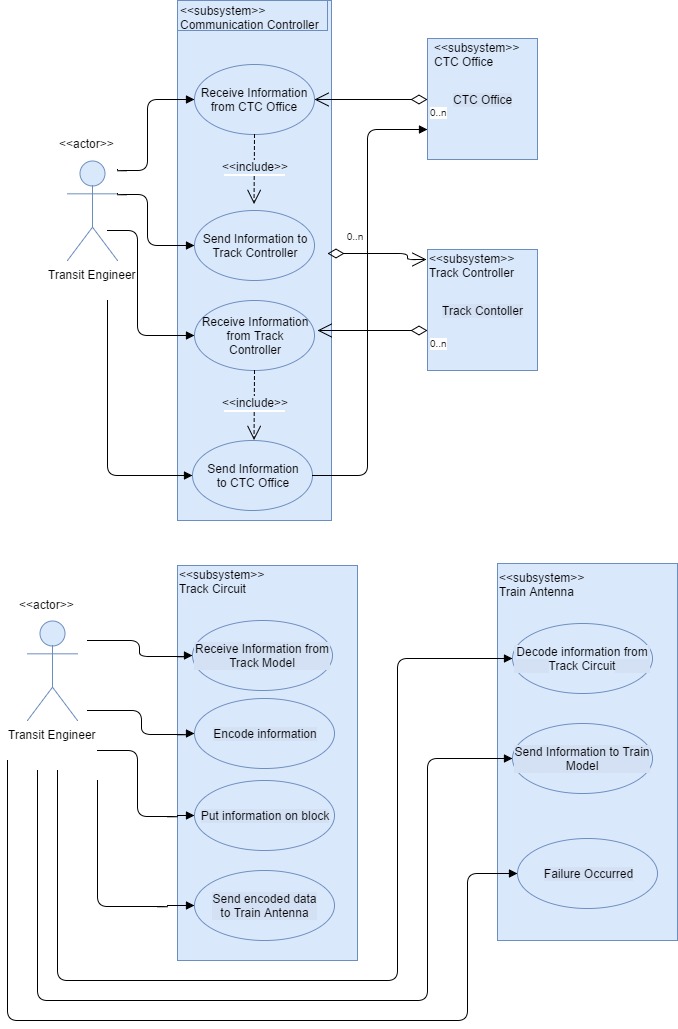
***Flip Switch***

1. User selects a switch from the switch list.
2. The switch viewer displays the switch's current position.
3. The user hits the Flip button.
4. The switch viewer and map display the new switch position.

***Follow Automatic Schedule***

1. User selects either Fixed Block or Moving Block automatic modes
2. In either mode, the UI controls for trains and blocks disable until the user re-selects Manual mode.
3. The map and displays update to show trains' movements as they follow the schedule.

**Communication:**



***Receive information from CTC Office and Send Information to Track Controller***

1. CTC Office sends information to Communication Controller, and the user is able to see that information was received via blue indicator light
2. Information received will be displayed in UI
3. User is able to see if the information that will be sent to the Track Controller is valid or not via red indicator light
4. Information to be sent will be displayed in UI

***Receive information from Track Controller and Send Information to CTC Office***

1. Track controller sends information to Communication Controller, and the user is able to see that information was received via blue indicator light
2. Information received will be displayed in UI
3. User is able to see if the information that will be sent to the CTC Office is valid or not via red indicator light
4. Information to be sent will be displayed in UI

***Receive information from Track Model***

1. Track Model sends information to Track Circuit, and the user is able to see that information was received via blue indicator light
2. User is able to see if information received is invalid by red indicator light
3. Information received will be displayed in UI

***Encode information***

1. Encoded data, in the form of a String of bits of 0s and 1s, will be displayed in UI

***Put information on block***

1. User is notified via red indicator light if information was not put on the block

***Send encoded data to Train Antenna***

1. User is notified via red indicator light if information was not sent to Train Antenna

***Decode information from Track Circuit***

1. User is notified via blue indicator light when information is received from Track Circuit
2. User is notified via red indicator light if information received is invalid
3. Received encoded String of bits is displayed to user in UI
4. Resulting decoded data is displayed to user in UI

***Send information to Train Model***

1. User is notified via red indicator light if information that will be sent to the Train Model is invalid

***Failure Occurred***

1. User pushes "**Report Failure**" button if invalid information will be sent to Train Model; button is only functional if data is invalid
2. Train Model is notified that vital information from Train Antenna is invalid

**Track Model:**



***Load New Track:***

1. User goes to "File -> Open"
2. User chooses Track Model Excel File from the computer files and clicks "Open"

***View Block Details***

1. User chooses "Red" or "Green" line from dropdown menu labeled "Line".
2. User chooses one of the given "Sections" from the dropdown menu labeled "Sections".
3. User chooses one of the given "Blocks" from the "Section" previously chosen in the dropdown menu labeled " Blocks".

***Or***

1. User chooses "Red" or "Green" line from the dropdown menu labeled "Line".
2. User chooses one of the given "Blocks" from the "Line" previously chosen in the dropdown menu labeled "Blocks"

***Change Temperature:***

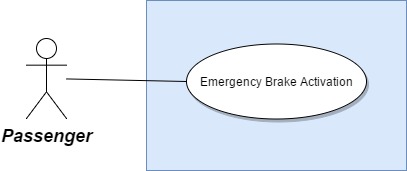
1. User inputs temperature into designated block next to actual temperature.
2. User hits the "Update" button.

*Note***:** Dialogue box will pop up to warn user of error if temperature is not between 0 and 100°F.

***Cause Failure:***

1. Follow the steps of ***View Block Details*** (This will be for choosing the block that you wish to mark with a failure).
2. Check any of the boxes labeled, "Break Rail", "Power Fail" or "Track Circuit Fail".
3. Hit the "Update" button.

**Train Model:**



***Emergency Brake Activation***

1. Passenger selects the "Emergency Brake" button.
2. The emergency brake will be engaged until the driver disengages it.

**Train Controller:**



***Set Speed:***

1. Driver enters desired speed in mph into the text box under the "Set Speed" heading.
2. Driver selects the "Set" button under the "Set Speed" heading.
   1. The speed will not change and an error will be displayed if the entered value exceeds the speed limit or an illegal character is entered.
3. The train will accelerate or decelerate to the new speed.

***Set Desired Temperature:***

1. Driver enters desired temperature in the "Set" text box under the "Temp" heading.
2. Driver selects the "Set" button under the "Temp" heading.

***Engage Emergency Brake:***

1. Driver selects the "Emergency Stop" button.
2. The emergency brake will remain engaged until the button is selected again.

***Open/Close Doors:***

1. Driver selects the "Open"/"Closed" radio button underneath either the "Left Doors" or "Right Doors" heading"
2. Door status will change to either "Open" or "Closed".
   1. If the train is in motion the doors will remain closed.

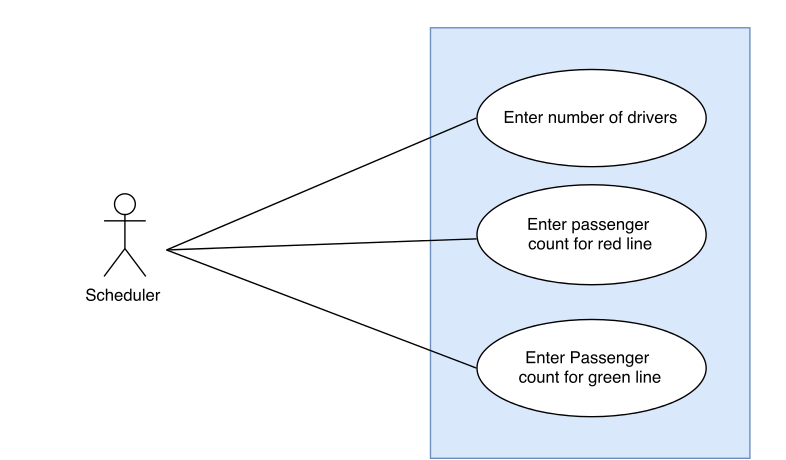
***Send Announcement:***

1. Driver enters message to send in the text box under the "Announcements" header.
2. Driver selects the "Send" button.
3. The message is sent to be displayed to passengers.

***Set Gain Values:***

1. Engineer enters the proportional gain in the "Kp" text box under the "Gain" header.
2. Engineer enters the integral gain in the "Ki" text box under the "Gain" header.
3. Engineer selects the "Set" button under the "Gain" header.
   1. The values will not be changed if an illegal character is entered.
4. The train will begin using the entered gain values.

**Moving Block Overlay:**



**Create Schedules:**

1. Scheduler enters number of drivers for one week.
2. System Validates input.
3. Scheduler enters the number of expected passengers for the red line.
4. System validates input.
5. Scheduler enters the number of expected passengers for the green line.
6. System validates input.
7. Scheduler clicks “Generate” button.
8. System loads track.
9. System calculates line run times.
10. System calculates schedules for drivers.
11. System calculates schedules for both lines.
12. System Generates red line schedule, green line schedule, and driver schedule and outputs them to screen

**Variation 1:** Scheduler enters invalid input for number of drivers.

* 1. Start at step 2.
  2. Scheduler enters input that is not a positive, non-zero, integer.
  3. Pop-up window comes up stating “Number of drivers must be a positive integer”
  4. Schedulers original input is erased.
  5. Continue at step 1.

**Variation 2:** Scheduler enters invalid input for number of passengers for the red line.

* 1. Start at step 4.
  2. Scheduler enters input that is not a positive, non-zero, integer.
  3. Pop-up window comes up stating “Number of passengers must be a positive integer”
  4. Schedulers original input is erased.
  5. Continue at step 4.

**Variation 3:** Scheduler enters invalid input for number of passengers for the green line.

* 1. Start at step 6.
  2. Scheduler enters input that is not a positive, non-zero, integer.
  3. Pop-up window comes up stating “Number of passengers must be a positive integer”
  4. Schedulers original input is erased.
  5. Continue at step 6.

**View Train Schedule:**

1. Generate Schedule.
2. Click Train Schedules Tab.
3. Choose Stop from the drop down menu.
4. System calculates the arrival times for stop.
5. System outputs the arrival time for the stop.

**View All Drivers Schedule:**

1. Generate Schedule.
2. Click Driver Schedule Tab.

**View Detailed Driver Schedule:**

1. View All Drivers Schedule.
2. Click on Driver ID that would like to view details for.
3. Click on details tab.

**Calculate safe stopping distance:**

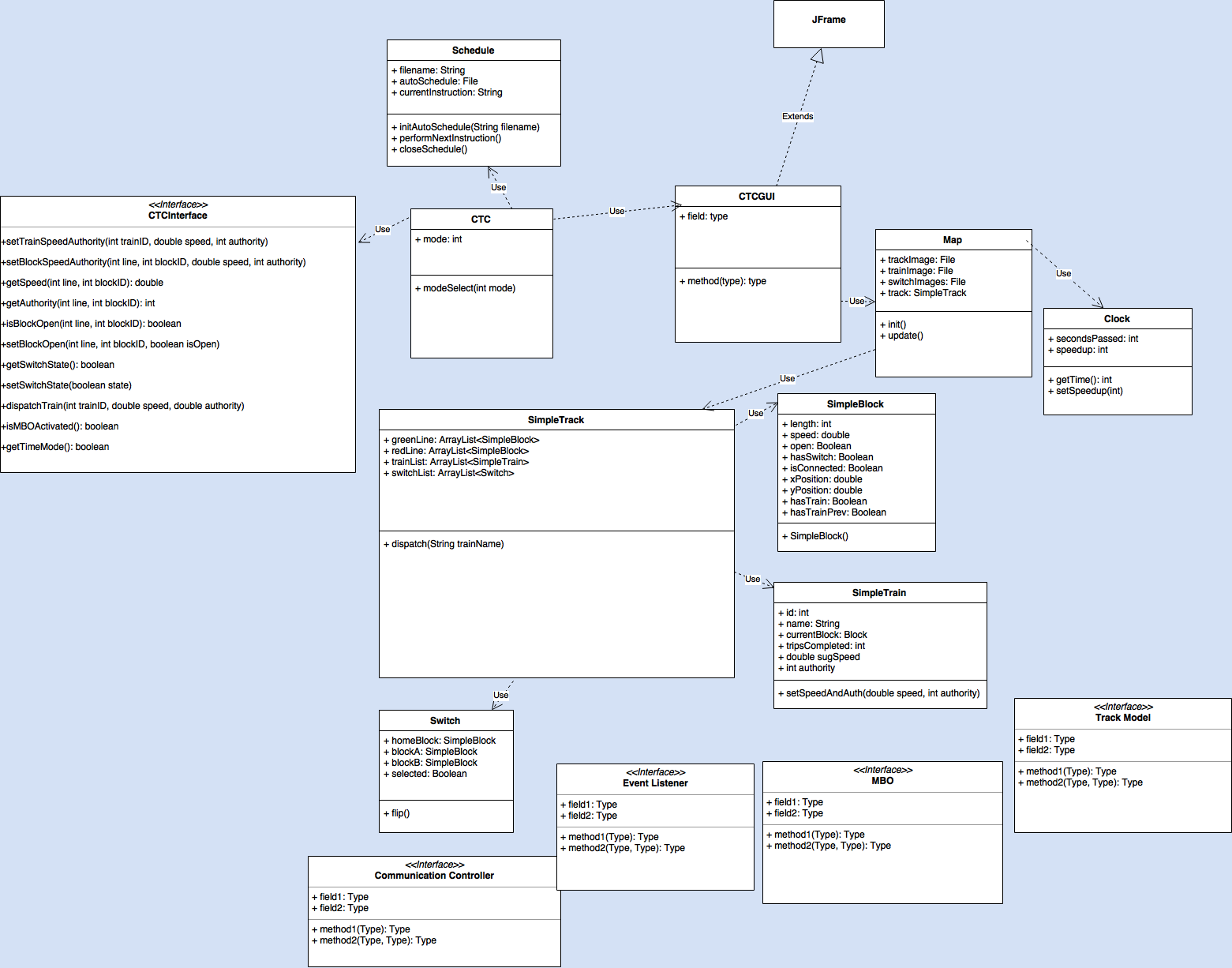
1. System loads track model.
2. System saves highest speed.
3. System calculates stopping distance with highest speed and breaking constant of 1.2 m/s2.

**Class Diagrams**

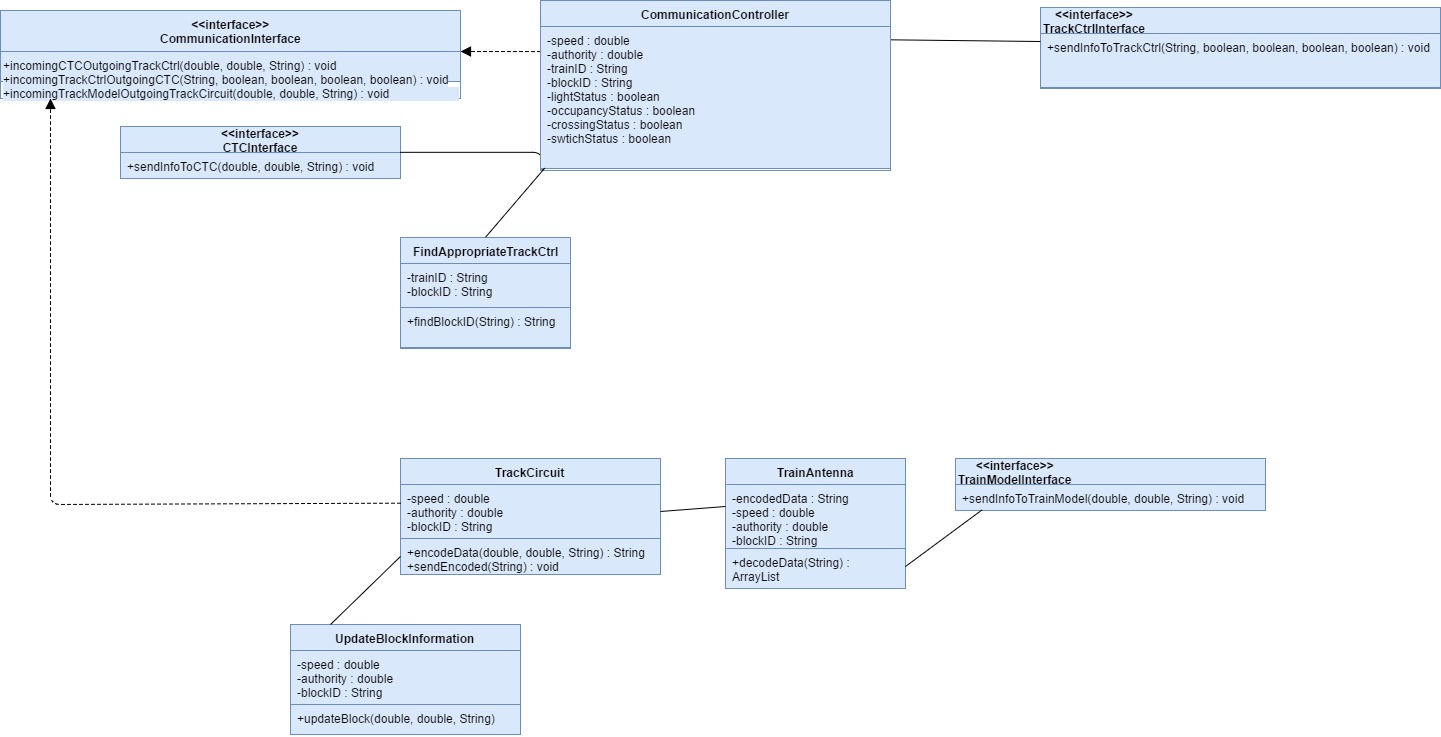
**Overall:**



**CTC:**



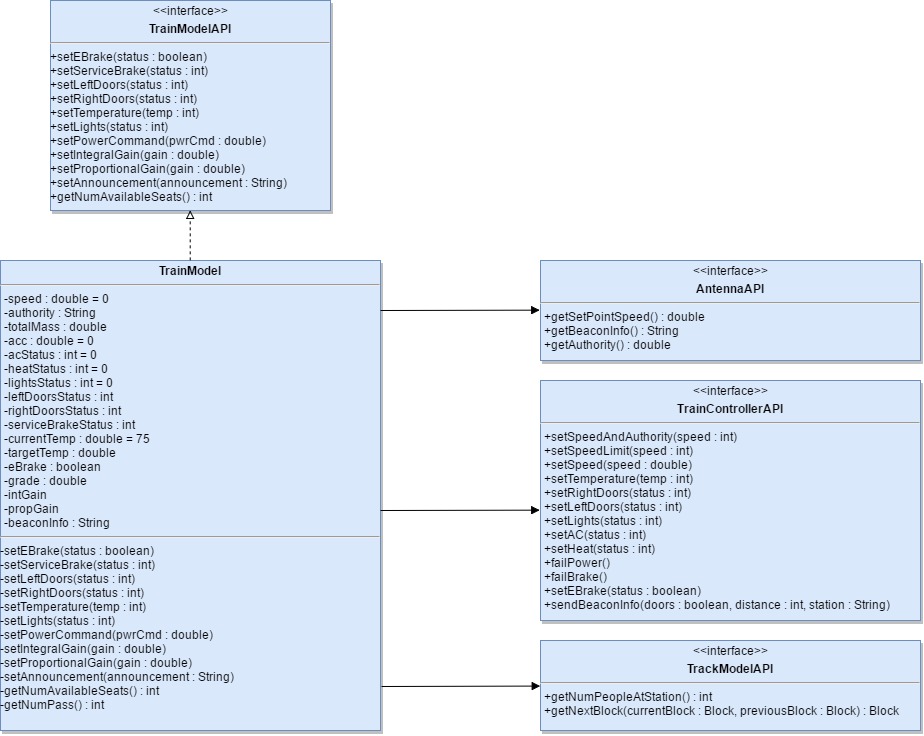
**Communication:**



**Track Model:**



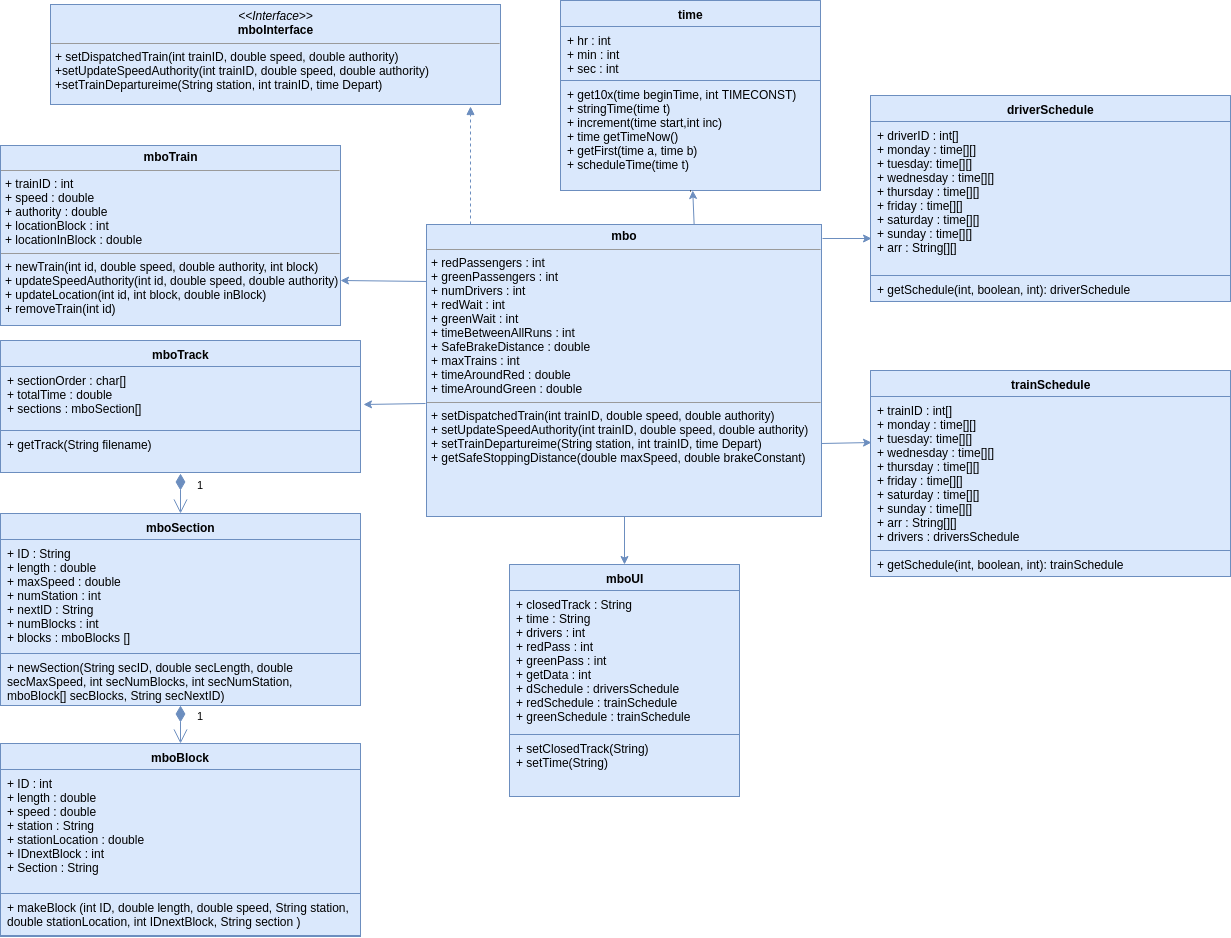
**Train Model:**



**Train Controller:**



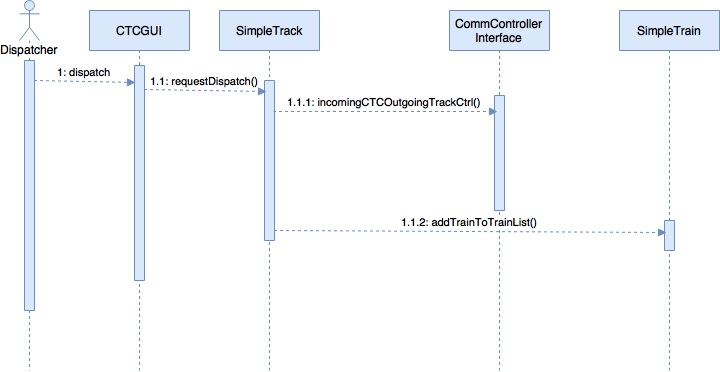
**Moving Block Overlay:**



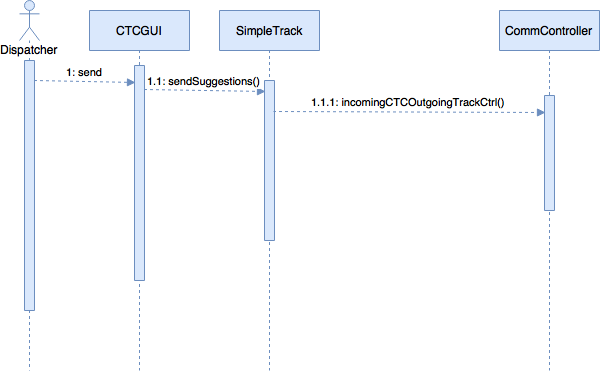
**Sequence Diagrams**

**CTC:**

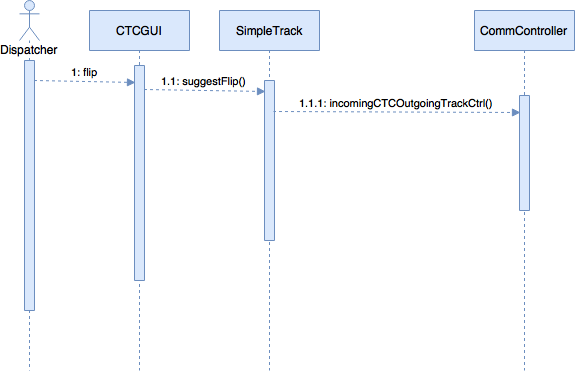
*Dispatch Train*



*Send Speed/Authority*

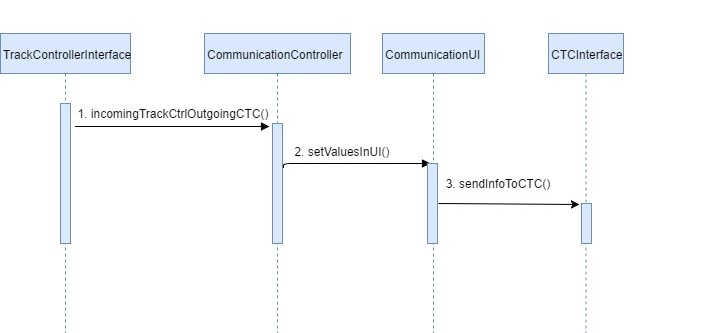


*Flip Switch*

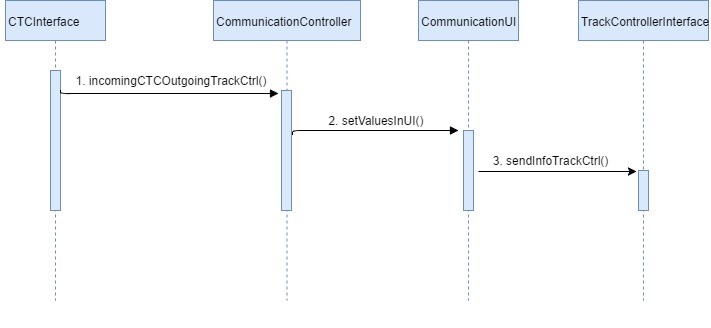


**Communication:**

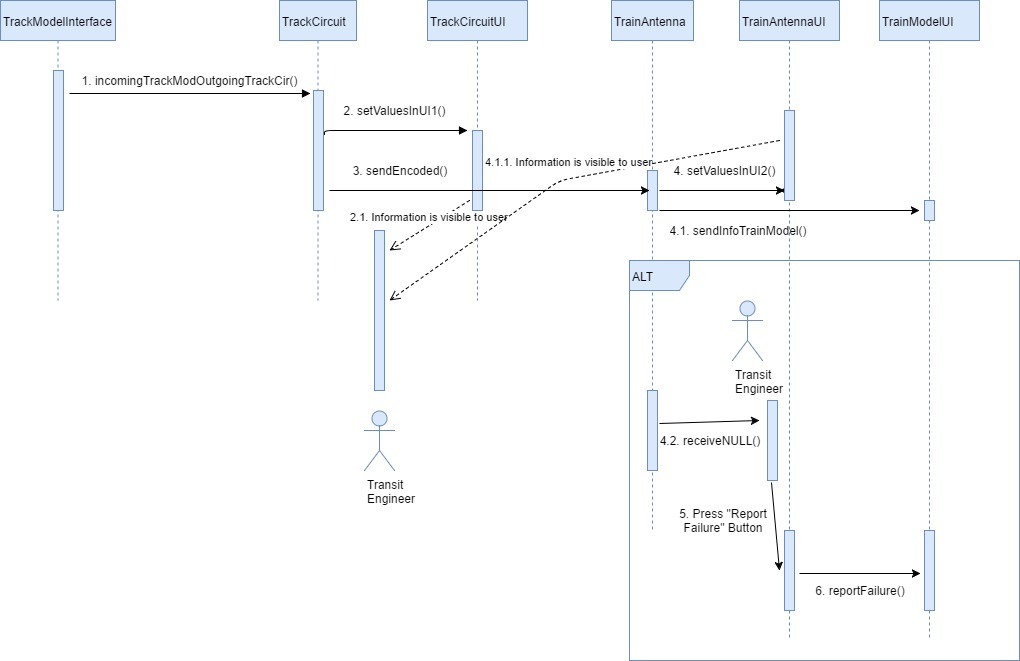
*Receive data from Track Controller and send data to CTC*



*Receive data from CTC and send data to Track Controller*



*Handle communication between Track Model and Train Model*



**Track Model:**

*Set Temperature*



*Set Failures*



*Load File*

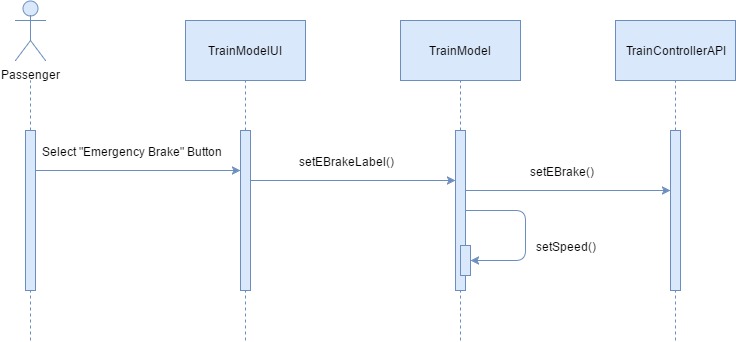


*View Block*



**Train Model:**

Emergency Brake Activation



**Train Controller:**

Set Speed



Engage Emergency Brake



Open/Close Doors



Send Announcement

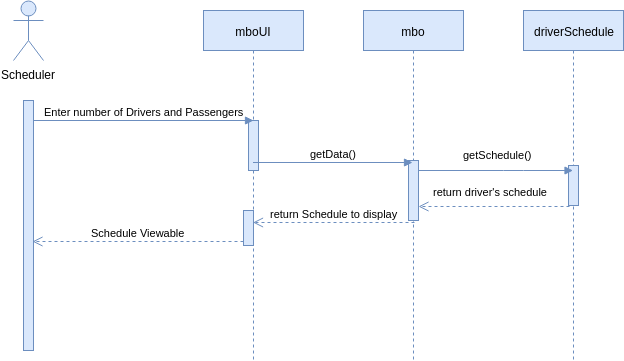


Set Gain Values

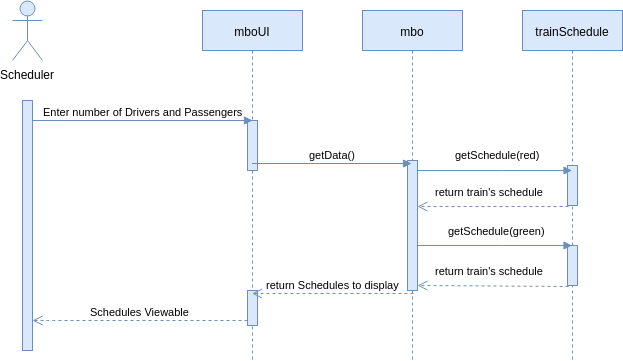


**Moving Block Overlay:**

Get Driver Schedule:



Get Train Schedules:

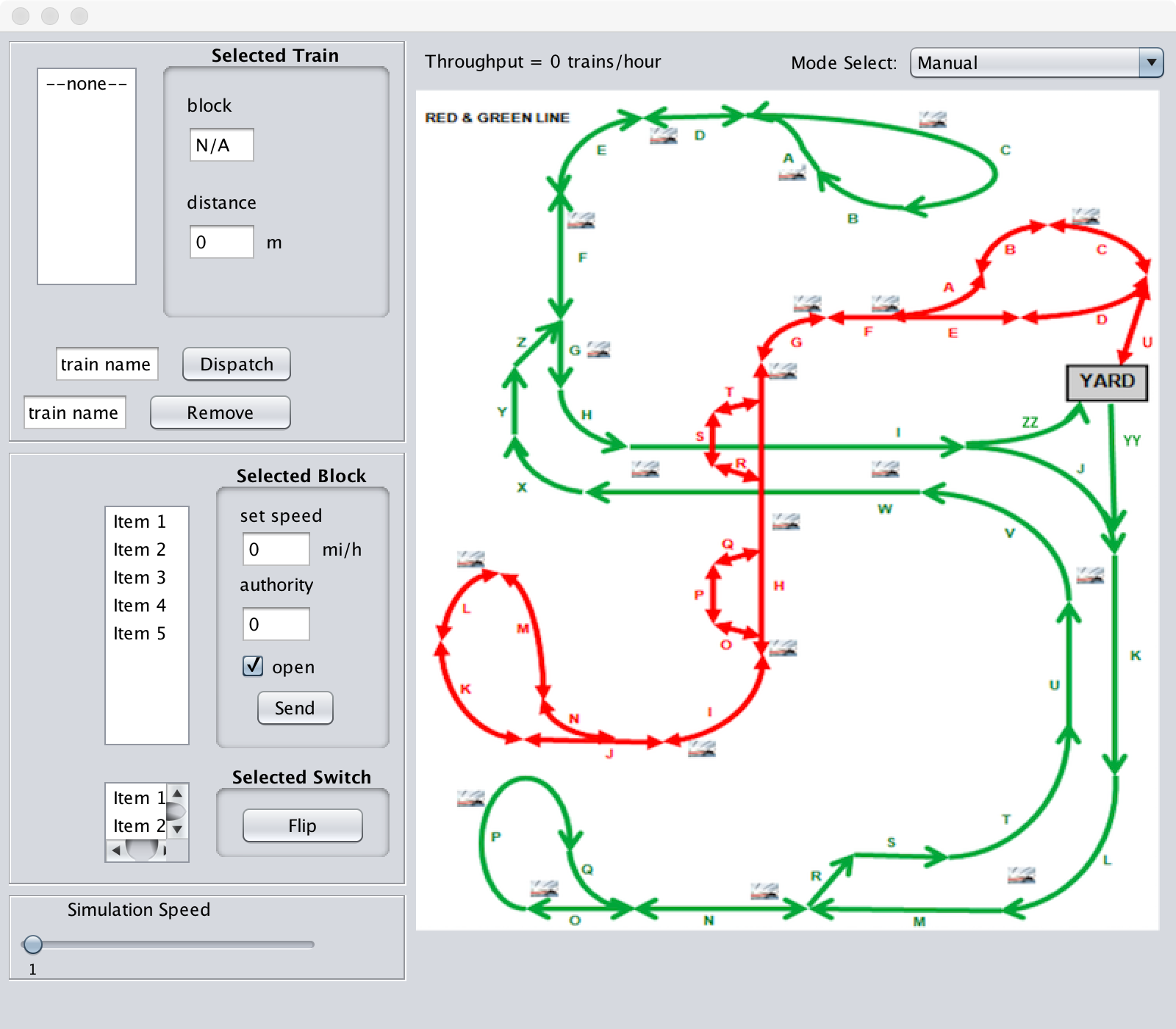


**Design Rationale**

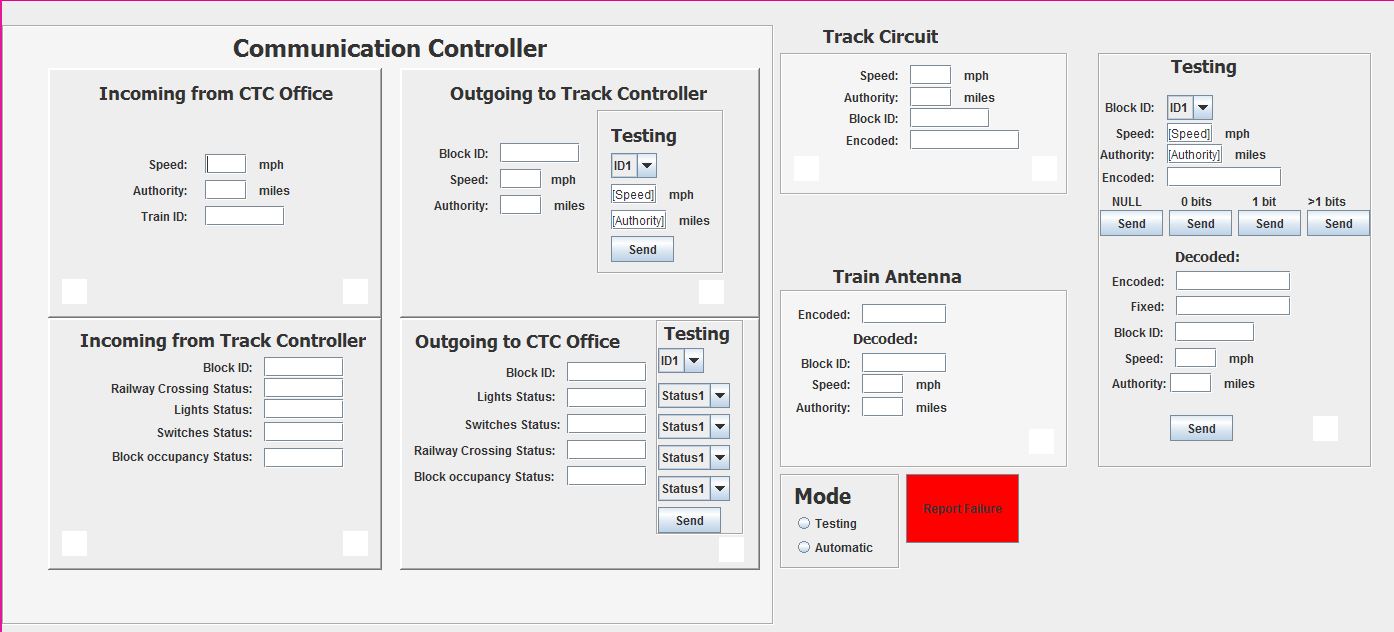
The design is modular with communication between modules using standard interfaces. The modular design was chosen because it allows each module to appear as a black box with expected inputs and outputs to other modules. Treating other modules as a black box simplifies the implementation because when the internal workings of a module change, the other modules remain unaffected as opposed to a waterfall of changes being made.

**UI**

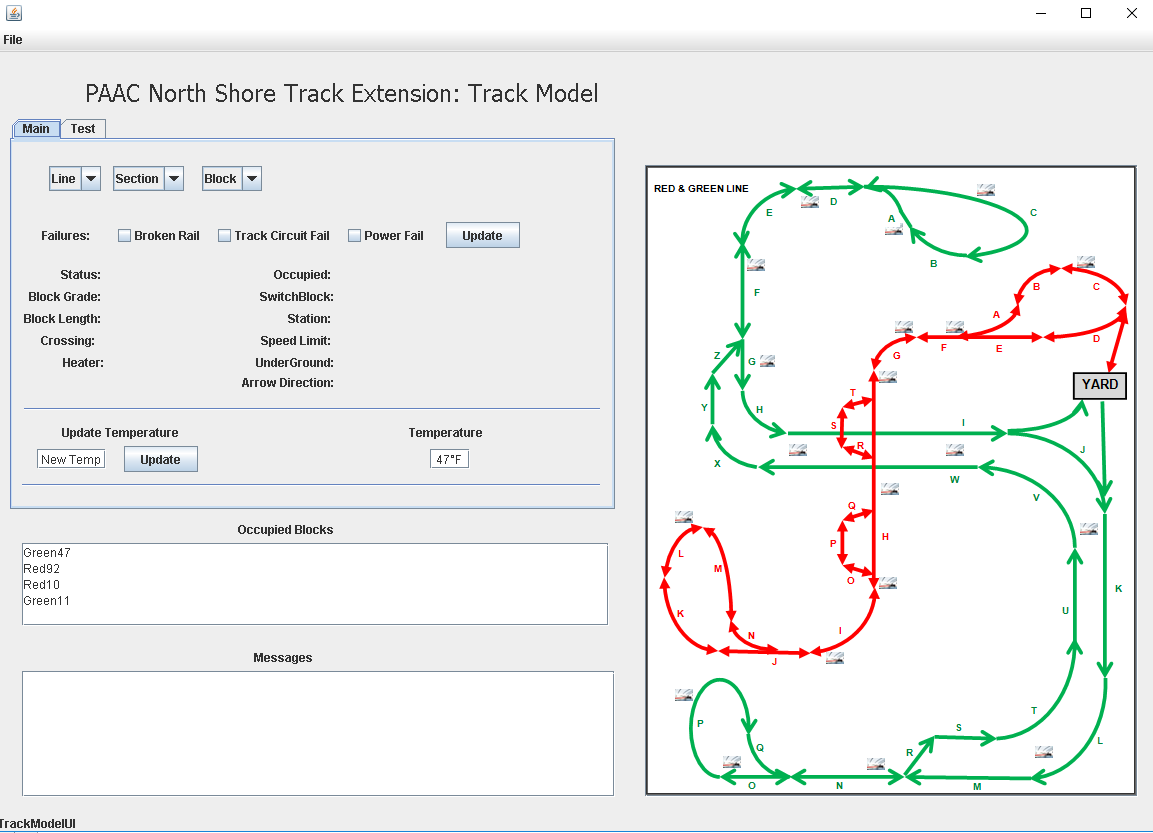
**CTC:**



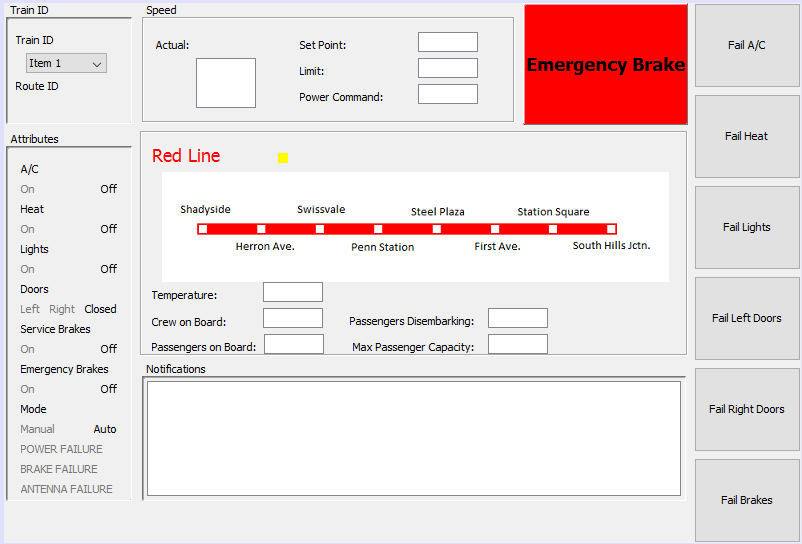
**Communication:**



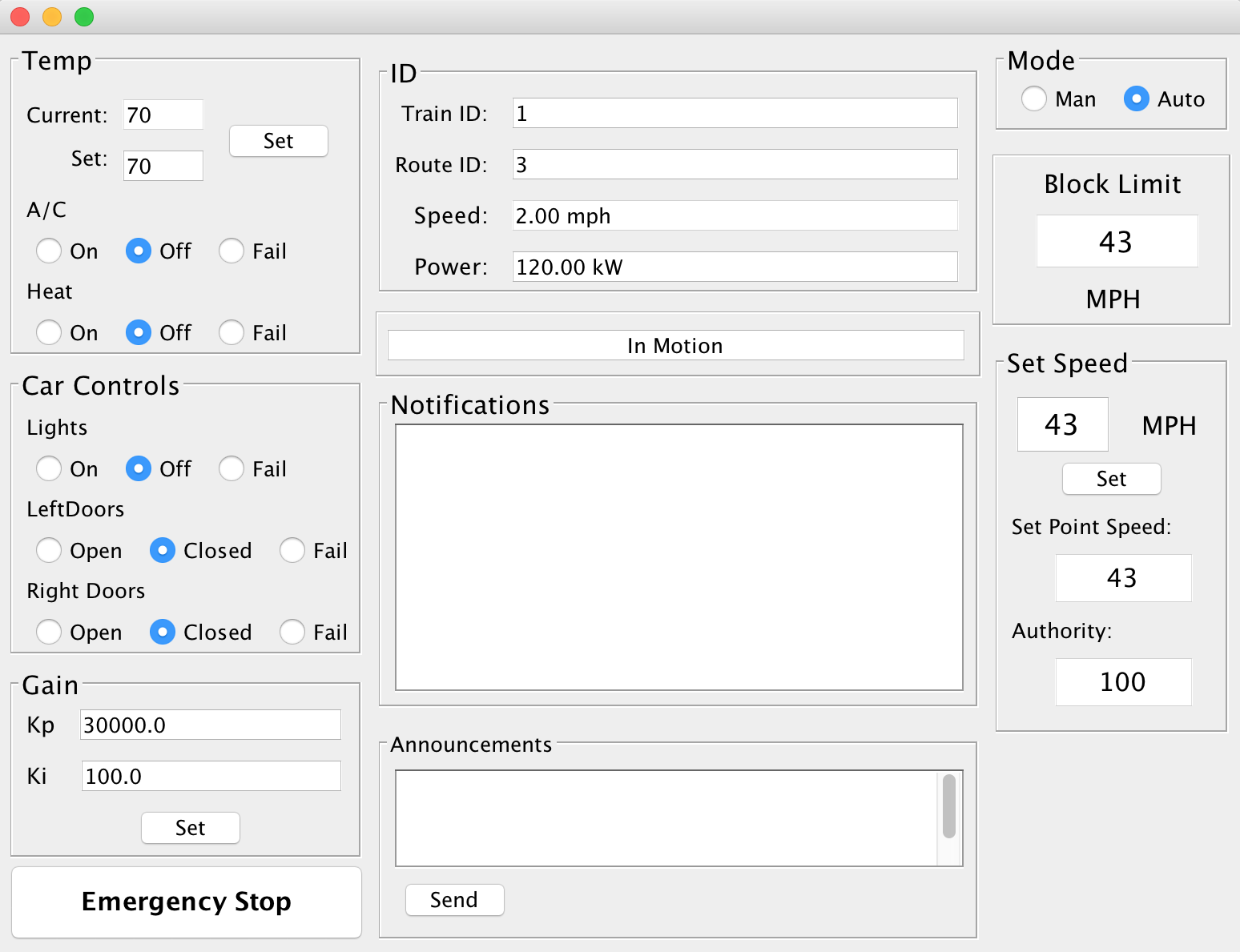
**Track Model:**



**Train Model:**



**Train Controller:**



**Moving Block Overlay:**

