Configuration Management

Mysterious Steam

1. CTC Module, Track Controller, Train Controller, Train Model, and Track Model created separately – 11/9/2012
2. Train Model added to Train Controller – 11/13/2012
   1. Train Controller can create and modify trains
3. Track Model added to Track Controller - 11/13/2012
   1. Track controller instantiates Track Model
4. Track Controller added to CTC Module – 11/13/2012
   1. CTC can create Track Controller
5. Train Controller and Train Module added to CTC Module – 11/16/2012
   1. CTC Module can now create trains through Track Controller and Train Controller
6. Train Movement added to Train Model – 11/23/2012
   1. Trains now moving on the track
7. Power and brake controls added to Train Controller - 11/30/2012
   1. Can now control train movement
8. Movement control added Track Model – 12/6/2012
   1. Track Model can choose correct path for trains
9. Switch Controls added to Track Controller – 12/7/2012
   1. Trains can now move across switches
10. Station Visits implemented in Train Controller and Track Controller – 12/11/2012
    1. Trains will stop at stations
11. Failure Modes and failure handling added to models and controllers – 12/12/2012
12. Miscellaneous changes made to modules – 12/13/2012

**Test Vectors:**

1. Create Train(Input: Train Controller Created; Output: Initial Values)
   1. CTC Office creates instance of Train Controller
   2. Train Controller creates instance of Train Model
   3. Check initial variables to see if the train was created correctly or not

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Caleb Dusenbery | Andrew Beers | Pass |  |

1. Create Panel(Input: Button Click; Output: GUI)
   1. CTC Office creates instance of Train Controller
   2. Train controller creates an instance of Train Model
   3. Train Controller sets initial values for Train Model
   4. Panel creates all buttons for the GUI
   5. GUI displayed

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Caleb Dusenbery | Andrew Beers | Pass |  |

1. Change Speed(Input: Power; Output: Train Model Variables)
   1. Train Model receives a change to power
   2. Train model processes new input power and calculates acceleration and velocity
   3. Train Model displays new acceleration, power, and velocity

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Caleb Dusenbery | Andrew Beers | Pass | Acceleration, and Power not displayed |

1. Decrease Speed(Input: Brake signal;Output: Acceleration)
   1. Train Model receives new brake value
   2. Train Model processes brake amount and calculates appropriate acceleration
   3. Train Model displays acceleration

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Caleb Dusenbery | Andrew Beers | Pass | No brakes, negative power |

1. Light Change(Input: Light value; Output: Light ON)
   1. Train Model receive new lights value
   2. Train Model shows light is on

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Caleb Dusenbery | Andrew Beers | Pass |  |

1. Passenger Change(Input: Passenger; Output: Passenger value)
   1. Train Model receives a new value for passengers
   2. Train Model displays correct amount of passengers

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Caleb Dusenbery | Andrew Beers | Pass |  |

1. Door Change(Input: Door value; Output: Door)
   1. Train Model gets new value for doors
   2. Train Model shows door open

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Caleb Dusenbery | Andrew Beers | Pass |  |

1. Simulate Failure(Input: Failure; Output: Failure Signaled)
   1. Train Model User clicks a failure option
   2. Signal Train controller of failure

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Caleb Dusenbery | Andrew Beers | Pass |  |

1. Emergency Brake (Passenger) (Input: Button Click; Output: Deceleration)
   1. Train Model User clicks Emergency Brake button
   2. Train Model uses maximum allowed deceleration
   3. Train Model displays maximum deceleration

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Caleb Dusenbery | Andrew Beers | Pass |  |

1. Crew Count (Input: Crew Number; Output: Crew Number)
   1. Train Controller sets crew number when creating Train Model instance
   2. Train Model displays correct crew number

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Caleb Dusenbery | Andrew Beers | Pass | 1 crew, driver |

1. Train Size (Input: Length, Width, and Height; Output: Length, Width, and Height)
   1. Train Controller sets Length, Width, and Height when creating Train Model instance
   2. Train Model displays correct Length, Width, and Height

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Caleb Dusenbery | Andrew Beers | Pass | Not displayed |

1. Train Cars (Input: Cars; Output: Cars)
   1. Train Controller sets number of cars number when creating Train Model instance
   2. Train Model displays correct number of cars

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Caleb Dusenbery | Andrew Beers | Pass | 1 car per train |

1. Train Mass (Input: Initial Variables, Passenger Count; Output: Mass)
   1. Train Controller sets initial variables when creating Train Model instance
   2. Train Model calculates mass based on initial variables and current passenger count
   3. Train Model displays correct mass

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Caleb Dusenbery | Andrew Beers | Pass |  |

1. Route Information (Input: Route information; Output: Route information)
   1. Train Controller sets Route information when creating Train Model instance
   2. Train Model displays correct Length, Width, and Height

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Caleb Dusenbery | Andrew Beers | Pass |  |

1. Temperature (Input: Temperature; Output: Temperature)
   1. Train Controller sets temperature
   2. Train Model displays correct temperature

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Caleb Dusenbery | Andrew Beers | Pass |  |

1. Current Position (Input: block length, acceleration; Output: distance)
   1. Train Model calculates current distance since the start of the block
   2. Train Model displays correct distance

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Caleb Dusenbery | Andrew Beers | Pass |  |

1. Track Information (Input: Track Model Variables, Output: Track Model Variables)
   1. Train Model gets track block information from Track Model
   2. Train Model displays correct information

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Caleb Dusenbery | Andrew Beers | Pass |  |