Low-Comotovation: Group Test Plan

Michael Ghaben

Contents

1 Introduction **1**
1.1 Document Identifier 1
1.2 Scope 1
1.3 System Overview and Key Features 1

2 Test Overview **2**
2.1 Master Test Schedule 2
2.2 Integrity Level Schema 2
2.3 Responsibilities 2
2.4 Tools, Techniques, and Metrics 3

3 Details **3**
3.1 Process 3
3.2 Test Documentation Requirements 3
3.3 Test Administration Requirements 4
3.4 Test Reporting Requirements 4

4 Track Module Test Plan **4**
4.1 Unit Tests 5
4.2 Integration Tests 9

5 Track Controller Test Plan **10**
5.1 Unit Tests 10
5.2 Integration Tests 11

6 CTC Test Plan **11**
6.1 Unit Tests 11
6.2 Integration Tests 11

7 MBO Test Plan **11**
7.1 Unit Tests 11
7.2 Integration Tests 11

8 Train Model Test Plan **11**
8.1 Unit Tests 11
8.2 Integration Tests 20

9 Train Controller Test Plan **20**
9.1 Unit Tests 20
9.2 Integration Tests 23

10 Changelog **25**

1 Introduction

1.1 Document Identifier

This document is the Low-Comotovation Group Test Plan for the design review phase of the Spring 2017 software engineering project.. In this document we detail the testing specifications, requirements, and procedures for the implementation and evaluation of testing procedures.

1.2 Scope

In this document a number of assumptions regarding the projects scope and lifecycle are made. Specifically:

1. Due to the short development cycle associated with this project, some non-negligble defects will likely persist

2. Under the iterative development methodology this group is following, the development of additional tests may be completed to address defects is expected

To better address these constraints, we will utilize a continuous integration methodology around iterative development and testing.So that we may ensure rapid development with minimal time overhead, we shall utilize continuous integration tools and methodology. Testing will be broadly divided into two categories: subsystem, and integration. Subsystem testing will be primarily focused on testing an individual system to ensure minimal functionality of a given subsystem and be done primarily independently by each group member. Integration testing will be designated as tests which require two or more modules functionality to satisfy requirements.

Specifically, it is anticipated that software defects will be found and require that testing which was not anticipated. As a result, we shall primarily focus on defining general tsts to be implemented and leave the exact testing requirements as implementation is completed. Additionally, will primarily focus on subsytems to be delivered to the end-user, the train company.

Examples of these subsystems would be the CTC module or the train controller, which are expected to be integrated into the final deliverable. Subsystems which are not examples of the final deliverables are the Train Model and Track Model subsystems, as they will ultimately be removed and replaced with the physical subsystems.

1.3 System Overview and Key Features

The purpose of the system under development is a to provide a train system for the Pittsburgh North Shore Rail system.

This system broadly consists of 6 subsystems:

1. Track Model - a physical model of a track to be used for testing

2. Train Model - a physical model of a train to be used for testing
3. CTC System - A CTC system to be implemented on the final train system
4. MBO - ???
5. Wayside System - A wayside for coordinating with all models

For a further discussion of the system, we refer the reader to the project reuqirements and discussion board.

2 Test Overview

The test organization is broadly divided into two sections: subsystem and integration testing. Subsystem tests are regarded as tests associated with only a single system at a time. Integration tests broadly refer to the tests of the integration of more than a single subsystem. In this view, testing is accomplished by each subsystem independently at the discresion of the individual developing the subsystem. Then, as members of the team develop their subsystem, each member shall attempt to integrate and develop tests for their integration. Due to the nature of continuous integration, tests are expected to be developed in parallel to the development of the program modules.

2.1 Master Test Schedule

The test schedul will be implemented as follows::

2.2 Integrity Level Schema

For this project, we utilize three integrity levels. The integrity levels for this project are as follows:

1. The lowest severity level. This is reserved for tasks which will ultimately not be passed onto the finished product and pose no threat to catastrophic failure.
2. This severity level is reserved for failures which may lead to errors in subsystem integration or incorrect information delivered to critical subsystems
3. The most severe integrity level. Is a vital system or otherwise threatens life or limb in an imlemented subsystem

2.3 Responsibilities

Michael Ghaben will be responsible for the integrity of the automated build system as well as the integrity of the master branch.

2.4 Tools, Techniques, and Metrics

To implement the test environment and better facilitate a continuous integration test environment, we utilize Travis-CI[1] with GitHub[2] integration with Slack[3] and Jupiter JUnit [4] integration. By utllizing these tools, we allow automated tests to be run remotely using the Maven[5]. This allows for rapid feedback into the developmental process, allowing for better integration and testing of the team. The usage of these tools creates feedback loop between implementation, testing, and integration, leading to significant productivity gains. In each case ,testing will be carried out remotely by the build system.

The system shall be quantitatively evaluated on percentage of percentage of test passing. Each subsystem shall be responsible for the determination of importance of testing individual components of their subsystem, with the exception of vital components.

3 Details

3.1 Process

A test shall be detailed in the following manner:

Table 1: Test Plan

Task	Test Design
Integrity Level hline Methods	What is the integrity level? How?
Inputs	What Inputs?
Outputs	What is a successful output?
Expected Completion	When will it be done?
Risks and Assumptions	What are you assuming?
Responsibility	Who are you?

In this, we expect to utilize both integration as well as unit tests. After each member pushes to GitHub on his or her respective branch, Travis-CI will provide regression testing on all unit and integration tests that have been implemented. To merge into master, all tests must be passing.

3.2 Test Documentation Requirements

Each test shall be documented using the above table as well as any auxillary information by whomever holds testing responsibilities. Each module shall have

[1] Travis-CI.org
[2] github.com
[3] slack.com
[4] junit.org
[5] maven.apache.org

a specified test plan for each module covering their individual component for testing. Each subsystem test plan shall detail unit tests for his or her own module. Additionally, integration testing will be accomplished in a similar fashon completed by the group.

Furthermore, the unit and integration testing procedure will be supplemented by functional testing. Each group member shall conduct user testing of each other module. During this time, the testing member will attempt to cause defective behavior at any level. These defects will be tracked via GitHub issues. This testing will occur weekly.

3.3 Test Administration Requirements

For a unit or integration test to be considered complete, it must successfully build on the build server utilizing the Maven build system. This ensures consistent repeatable builds to attempt to ensure the clients functional requirements will be met.

Additionally, for functional testing it is expected that each group member submits either a bug report via GitHub issues. Should a group member fail to find any defects and register them, he or she must challenge Professor Profeta to find a defect at the next class meeting. If the professor discovers a defect, the group member(s) who failed to find defects owe the other group members pizza. It is hoped that this procedure will lead to people finding more defects.

3.4 Test Reporting Requirements

Each written unit and integrationtest report will be provided by the Maven automated build system.

To document user testing (e.g. by working with the user interface and attempting to find defects in that manner), a developmer may supplement this test report with the following syntax to automatically document the bug utilizng the following syntax:

```
/**
* @bug < Descriptive message >
*/
```

This will lead to documentation of the defect in the autmatically generated Doxygen documentation. Note that this should be used for defects which are not necessarily covered under tests at a given time. By utilizing this process, an iterative cycle of development these defects may be tracked and inclusion in later tests to ensure proper functionality.

4 Track Module Test Plan

Author: Michael Ghaben

4.1 Unit Tests

Table 2: CSV Reading Test Plan

Task	Test Design
Integrity Level	1
Methods	Evaluate the readCSV function
Inputs	The files redline.csv
Outputs	The track model successfully reading the redline csv files
Expected Completion	March 20, 2017
Risks and Assumptions	Both redline and greenline have been properly input to the files
Responsibility	Track Model

Table 3: CSV Reading Test Plan

Task	Test Design
Integrity Level	1
Methods	Evaluate the readCSV function on green line
Inputs	The files greenline.csv
Outputs	The track model successfully reading the redline csv files
Expected Completion	March 15, 2017
Risks and Assumptions	Both redline and greenline have been properly input to the csv files
Responsibility	Track Model

Table 4: Test Switch Root Node Reading

Task	Test Design
Integrity Level	2
Methods	Evaluate the switch nodes association in TrackModel
Inputs	The file redline.csv
Outputs	The track model successfully holding the root nodes in the rootMap
Expected Completion	March 15, 2017
Risks and Assumptions	Both redline and greenline have been properly input to the csv files
Responsibility	Track Model

Table 5: Test Switch Root Node Reading

Task	Test Design
Integrity Level	2
Methods	Evaluate the switch nodes association in TrackModel
Inputs	The file greenline.csv
Outputs	The track model successfully holding the root nodes in the rootMap
Expected Completion	March 15, 2017
Risks and Assumptions	Both redline and greenline have been properly input to the csv files
Responsibility	Track Model

Table 6: Test Switch Node Leaf Reading

Task	Test Design
Integrity Level	2
Methods	Evaluate the leaf nodes association in TrackModel
Inputs	The file redline.csv
Outputs	The track model successfully holding the proper refererences in the Block object set by rootMap
Expected Completion	March 15, 2017
Risks and Assumptions	Both redline and greenline have been properly input to the csv files
Responsibility	Track Model

Table 7: Test Switch Node Leaf Reading

Task	Test Design
Integrity Level	2
Methods	Evaluate the leaf nodes association in TrackModel
Inputs	The file greenline.csv
Outputs	The track model successfully holding the proper refererences to the leaf nodes in the Block object set by rootMap
Expected Completion	March 15, 2017
Risks and Assumptions	Both redline and greenline have been properly input to the csv files
Responsibility	Track Model

Table 8: Test Switch Node Leaf Reading

Task	Test Design
Integrity Level	2
Methods	Evaluate the switching functionality in the red line
Inputs	The file redline.csv
Outputs	The proper block given a non-default switch state
Expected Completion	March 15, 2017
Risks and Assumptions	Both redline and greenline have been properly input to the csv files
Responsibility	Track Model

Table 9: Test nextBlockForward() Red Line

Task	Test Design
Integrity Level	2
Methods	Evaluate the functionality of the nextBlockForward() function on the redline
Inputs	The file redline.csv
Outputs	The proper block given a switch on the red line
Expected Completion	March 15, 2017
Risks and Assumptions	Both redline and greenline have been properly input to the csv files
Responsibility	Track Model

Table 10: Test nextBlockForward() Green Line

Task	Test Design
Integrity Level	2
Methods	Evaluate the functionality of the nextBlockForward() function on the green line
Inputs	The file greenline.csv
Outputs	The proper block given a switch on the green line
Expected Completion	March 15, 2017
Risks and Assumptions	Both redline and greenline have been properly input to the csv files
Responsibility	Track Model

Table 11: Test nextBlockBackward() Red Line

Task	Test Design
Integrity Level	2
Methods	Evaluate the functionality of the nextBlockBackward() function on the red line
Inputs	The file redline.csv
Outputs	The proper block given a switch on the red line
Expected Completion	March 15, 2017
Risks and Assumptions	Both redline and greenline have been properly input to the csv files
Responsibility	Track Model

Table 12: Test nextBlockBackward() Green Line

Task	Test Design
Integrity Level	2
Methods	Evaluate the functionality of the nextBlockBackward() function on the red line
Inputs	The file redline.csv
Outputs	The proper block given a switch on the red line
Expected Completion	March 15, 2017
Risks and Assumptions	Both redline and greenline have been properly input to the csv files
Responsibility	Track Model

Table 13: Test nextBlockBackward() Secondary Switch Conditions Red Line

Task	Test Design
Integrity Level	2
Methods	Evaluate the functionality of the nextBlockBackward() function on the red line under the alternate switch functionality
Inputs	The file redline.csv
Outputs	The proper block given a switch on the red line
Expected Completion	March 15, 2017
Risks and Assumptions	Both redline and greenline have been properly input to the csv files
Responsibility	Track Model

Table 14: Test nextBlockBackward() Secondary Switch Conditions Green Line

Task	Test Design
Integrity Level	2
Methods	Evaluate the functionality of the nextBlockBackward() function on the red line under the alternate switch functionality
Inputs	The file greenline.csv
Outputs	The proper block given a switch on the red line
Expected Completion	Before Half-Life 3
Risks and Assumptions	Both redline and greenline have been properly input to the csv files
Responsibility	Track Model

4.2 Integration Tests

Table 15: Test Track Controller Switching

Task	Test Design
Integrity Level	2
Methods	Evaluate the functionality of the track controller to switch a switch state
Inputs	The files redline.csv and greenline.csv
Outputs	The proper block given a switch on the green line and the MBO switching the switch
Expected Completion	April 1, 2017
Risks and Assumptions	Both redline and greenline switches are able to be toggled successfully
Responsibility	Track Model

Table 16: Test Track Controller Switching

Task	Test Design
Integrity Level	2
Methods	Evaluate the functionality of the track controller to switch a switch state
Inputs	The files redline.csv and greenline.csv
Outputs	The proper block given a switch on the green line and the MBO switching the switch
Expected Completion	April 1, 2017
Risks and Assumptions	Both redline and greenline switches are able to be toggled successfully
Responsibility	Track Model

5 Track Controller Test Plan

5.1 Unit Tests

Table 17: Load PLC Program via Browse Button

Task	Test Design
Integrity Level	1
Methods	Evaluate the tryPLC() function and Browse Button
Inputs	PLC file selected via 'Browse' button
Outputs	'Success' notification displayed
Expected Completion	After initialization, but before any trains dispatched
Risks and Assumptions	PLC files exist in src/main/java/WaysideController/PLCResources (for automated testing, other external files may be used)
Responsibility	Wayside Controller

5.2 Integration Tests

6 CTC Test Plan

6.1 Unit Tests

6.2 Integration Tests

7 MBO Test Plan

7.1 Unit Tests

7.2 Integration Tests

8 Train Model Test Plan

8.1 Unit Tests

Table 18: Base Test A: Compute Velocity of train at rest with Power command of 100kW

Task	Test Design
Integrity Level	2
Methods	Apply power command to train and compute velocity
Inputs	Power Command input and ”Start Test” button is pressed
Outputs	Velocity greater than 0 MPH will be produced
Expected Completion	Test to be performed upon completion of complete submodule. Expected date: March 24th
Risks and Assumptions	The power command should be 100kW for this base case. Assumption will be made that for base test train starts with 0 velocity
Responsibility	Train Model

Table 19: Repeat Base Test A with Power command greater than 100,000W

Task	Test Design
Integrity Level	2
Methods	Apply higher power command to train and compute velocity
Inputs	Power Command input and ”Start Test” button is pressed
Outputs	Velocity greater than base case A will be produced
Expected Completion	Test to be performed upon completion of complete submodule. Expected date: March 24th
Risks and Assumptions	The power command should be a positive value greater than 100kW
Responsibility	Train Model

Table 20: Repeat base test A with Power command less than 100,000W

Task	Test Design
Integrity Level	2
Methods	Apply lower power command to train and compute velocity
Inputs	Power Command input and "Start Test" button is pressed
Outputs	Velocity lower than base case A will be produced
Expected Completion	Test to be performed upon completion of complete submodule. Expected date: March 24th
Risks and Assumptions	The power command should be a positive value smaller than 100kW
Responsibility	Train Model

Table 21: Repeat base test A with grade of 3%

Task	Test Design
Integrity Level	2
Methods	Increase grade to 3% and compute velocity
Inputs	Grade set to 3% and "Start Test" button is pressed
Outputs	Velocity lower than base case A will be produced
Expected Completion	Test to be performed upon completion of complete submodule. Expected date: March 24th
Risks and Assumptions	The power command should be equal to 100kW and grade will be set to 3%
Responsibility	Train Model

Table 22: Repeat base test A with grade of -3%

Task	Test Design
Integrity Level	2
Methods	Decrease grade to -3% and compute velocity
Inputs	Grade set to -3% and "Start Test" button is pressed
Outputs	Velocity greater than base case A will be produced
Expected Completion	Test to be performed upon completion of complete submodule. Expected date: March 24th
Risks and Assumptions	The power command should be equal to 100kW and grade will be set to -3%
Responsibility	Train Model

Table 23: Repeat base test A with 150 passengers added

Task	Test Design
Integrity Level	2
Methods	Increase passenger count to 150 and compute velocity
Inputs	Number of passengers = 150 and ”Start Test” button is pressed
Outputs	Velocity smaller than base case A will be produced
Expected Completion	Test to be performed upon completion of complete submodule. Expected date: March 24th
Risks and Assumptions	The power command should be equal to 100k0W and 150 passengers will be added onboard the train
Responsibility	Train Model

Table 24: Base Case B: Compute Velocity of train at 25MPH with Power command of 100kW

Task	Test Design
Integrity Level	2
Methods	Apply Power command of 100kW and compute new velocity
Inputs	Power Command set to 100kW and ”Start Test” button is pressed
Outputs	Velocity larger than 25MPH will be produced
Expected Completion	Test to be performed upon completion of complete submodule. Expected date: March 24th
Risks and Assumptions	The power command should be equal to 100kW
Responsibility	Train Model

Table 25: Repeat Base Case B with power command of 0W

Task	Test Design
Integrity Level	2
Methods	Apply Power command of 0W and compute new velocity
Inputs	Power Command set to 0W and ”Start Test” button is pressed
Outputs	Velocity smaller than 25MPH will be produced
Expected Completion	Test to be performed upon completion of complete submodule. Expected date: March 24th
Risks and Assumptions	The power command should be equal to 0W
Responsibility	Train Model

Table 26: Repeat Base Case B with power command less than 0W

Task	Test Design
Integrity Level	2
Methods	Apply negative power command and compute new velocity
Inputs	Power Command set to -100kW and "Start Test" button is pressed
Outputs	Invalid input message will appear
Expected Completion	Test to be performed upon completion of complete submodule. Expected date: March 24th
Risks and Assumptions	Power command must be positive for all possible cases

Table 27: Repeat Base Case B with power command greater than 120kW

Task	Test Design
Integrity Level	2
Methods	Apply power command above max and compute new velocity
Inputs	Power Command set to 150kW and "Start Test" button is pressed
Outputs	Speed will remain 25MPH as there is no more power available
Expected Completion	Test to be performed upon completion of complete submodule. Expected date: March 24th
Risks and Assumptions	If power exceeds max, the velocity stays the same

Table 28: Repeat Base Case B but apply Service brakes

Task	Test Design
Integrity Level	3
Methods	Apply service brakes and compute new velocity
Inputs	Service brakes engaged and "Start Test" button is pressed
Outputs	Service brake status switched to ON Power Command set to 0 Train speed decreased to lower than 25 MPH
Expected Completion	Test to be performed upon completion of complete submodule. Expected date: March 24th
Risks and Assumptions	Service brake will automatically override power command to 0W

Table 29: Repeat Base Case B but apply Emergency brakes

Task	Test Design
Integrity Level	3
Methods	Apply Emergency brakes and compute new velocity
Inputs	Emergency brakes engaged and "Start Test" button is pressed
Outputs	Emergency brake status switched to ON Power Command set to 0 Train speed decreased to lower than 25 MPH Train speed also lower than service brake test case
Expected Completion	Test to be performed upon completion of complete submodule. Expected date: March 24th
Risks and Assumptions	Service brake will automatically override power command to 0W

Table 30: Activate engine failure on moving train

Task	Test Design
Integrity Level	2
Methods	Toggle engine failure status to on
Inputs	Radio button for engine failure set to ON
Outputs	Engine Failure status switched to ON Power Command set to 0 Train speed decreased to 0 MPH Service brake status set to ON
Expected Completion	Test to be performed upon completion of complete submodule. Expected date: April 5th
Risks and Assumptions	Service brake will automatically activate on failure

Table 31: Activate Signal failure on moving train

Task	Test Design
Integrity Level	2
Methods	Toggle signal failure status to on
Inputs	Radio button for signal failure set to ON
Outputs	Signal Failure status switched to ON Power Command set to 0 Train speed decreased to 0 MPH Service brake status set to ON
Expected Completion	Test to be performed upon completion of complete submodule. Expected date: April 5th
Risks and Assumptions	Service brake will automatically activate on failure

Table 32: Activate Brake failure on moving train

Task	Test Design
Integrity Level	2
Methods	Toggle brake failure status to on
Inputs	Radio button for brake failure set to ON
Outputs	Brake Failure status switched to ON Power Command set to 0 Train speed decreased to 0 MPH Emergency brake status set to ON
Expected Completion	Test to be performed upon completion of complete submodule. Expected date: April 5th
Risks and Assumptions	Emergency brake will activate on failure in service brakes

Table 33: Open doors on moving train

Task	Test Design
Integrity Level	2
Methods	Open left side doors on moving train
Inputs	Radio button for left doors set to OPEN
Outputs	Invalid action pop-up
Expected Completion	Test to be performed upon completion of complete submodule. Expected date: April 5th
Risks and Assumptions	Doors will not open while train is in motion

Table 34: Open doors on non-moving train

Task	Test Design
Integrity Level	2
Methods	Open left side doors on non-moving train
Inputs	Radio button for left doors set to OPEN
Outputs	Left door status on Train model changes to OPEN
Expected Completion	Test to be performed upon completion of complete submodule. Expected date: April 5th
Risks and Assumptions	Left and Right doors can both be opened at the same time but opening is independent

Table 35: Power Command applied to non-moving train with open doors

Task	Test Design
Integrity Level	2
Methods	Open left door, then apply power of 100kW
Inputs	Radio button for left doors set to OPEN Power command Set to 100kW "Start Test" button pressed
Outputs	Error message pop-up
Expected Completion	Test to be performed upon completion of complete submodule. Expected date: April 5th
Risks and Assumptions	Train can not move if doors are open

Table 36: Power Command applied to non-moving train with Failure status

Task	Test Design
Integrity Level	2
Methods	Engage any failure, then apply power of 100kW
Inputs	Radio button for Engine Failure set to ON Power command Set to 100kW "Start Test" button pressed
Outputs	Error message pop-up
Expected Completion	Test to be performed upon completion of complete submodule. Expected date: April 5th
Risks and Assumptions	Train can not move if failures are present

Table 37: Power Command applied to non-moving train with engaged brakes

Task	Test Design
Integrity Level	2
Methods	Engage either brake, then apply power of 100kW
Inputs	Service Brakes engaged Power command Set to 100kW "Start Test" button pressed
Outputs	Error message pop-up
Expected Completion	Test to be performed upon completion of complete submodule. Expected date: April 5th
Risks and Assumptions	Train can not move if brakes are engaged

Table 38: Interior Light Test

Task	Test Design
Integrity Level	1
Methods	Lights turned on in test console
Inputs	Radio button for lights set to ON "Start Test" button pressed
Outputs	Interior Lights set to ON
Expected Completion	Test to be performed upon completion of complete submodule. Expected date: April 5th
Risks and Assumptions	Lights can be on at any time.

Table 39: Train Temperature set to 60F Thermostat set to 65F

Task	Test Design
Integrity Level	1
Methods	Set intial temp to 60F, Set Thermostat to 65F
Inputs	Train Temperature set to 60F Thermostat set to 65F "Start Test" button pressed
Outputs	Heater set to ON AC set to OFF Temperature increases to 65F
Expected Completion	Test to be performed upon completion of complete submodule. Expected date: April 5th
Risks and Assumptions	Heater and AC can not be on at the same time.

Table 40: Train Temperature set to 60F Thermostat set to 60F

Task	Test Design
Integrity Level	1
Methods	Set intial temp to 60F, Set Thermostat to 60F
Inputs	Train Temperature set to 60F Thermostat set to 60F "Start Test" button pressed
Outputs	Heater set to OFF AC set to OFF Temperature does not change
Expected Completion	Test to be performed upon completion of complete submodule. Expected date: April 5th
Risks and Assumptions	Temperature can only change if heat or AC is on No heat loss due to windows open

Table 41: Train Temperature set to 60F Thermostat set to 55F

Task	Test Design
Integrity Level	1
Methods	Set intial temp to 60F, Set Thermostat to 55F
Inputs	Train Temperature set to 60F Thermostat set to 55F "Start Test" button pressed
Outputs	Heater set to OFF AC set to ON Temperature decreases to 55F
Expected Completion	Test to be performed upon completion of complete submodule. Expected date: April 5th
Risks and Assumptions	Heater and AC can not be on at the same time.

Table 42: Integration test With train Controller

Task	Test Design
Integrity Level	1
Methods	Repeat above tests but receiving values from Train Controller
Inputs	Power Command, Utility statuses, Brake Statuses
Outputs	Outputs should reflect similar results as the test cases above
Expected Completion	Test to be performed upon completion of integration with train controller. Expected date: April 7th
Risks and Assumptions	Whether inputs come from train controller or test console results should be the same

Table 43: Integration test With Track Model

Task	Test Design
Integrity Level	1
Methods	Read in values for current block's grade for calculations
Inputs	Request for current grade to track Model
Outputs	If successful a grade will be returned to train model
Expected Completion	Test to be performed upon completion of integration with Track Model. Expected date: April 7th
Risks and Assumptions	Track model will send grade upon entrance to block

Table 44: Integration test With MBO

Task	Test Design
Integrity Level	1
Methods	Testing communication between MBo and Train model
Inputs	Request for current location from MBO
Outputs	If successful a location will be sent to the MBO
Expected Completion	Test to be performed upon completion of integration with MBO. Expected date: April 7th
Risks and Assumptions	Train model will periodically update MBO with location

8.2 Integration Tests

9 Train Controller Test Plan

Author: Andrew Lendacky

9.1 Unit Tests

Table 45: UI Elements Disabled in Automatic Mode

Task	Test Design
Integrity Level	2
Methods	Checks to see if the desired UI elements are disabled
Inputs	The variable 'inAutomaticMode'
Outputs	All the desired elements are disabled.
Expected Completion	When the system is in Automatic mode
Risks and Assumptions	The desired elements are known
Responsibility	Train Controller

Table 46: System is in Manual Mode

Task	Test Design
Integrity Level	1
Methods	Compares 'inManualMode' and 'inAutomaticMode'
Inputs	'inManualMode' and 'inAutomaticMode', which are booleans
Outputs	'inManualMode' is true and 'inAutomaticMode' is false
Expected Completion	When the system is switched to Manual mode
Risks and Assumptions	none
Responsibility	Train Controller

Table 47: System is in Automatic Mode

Task	Test Design
Integrity Level	1
Methods	Compares 'inManualMode' and 'inAutomaticMode'
Inputs	'inManualMode' and 'inAutomaticMode, which are booleans'
Outputs	'inManualMode' is false and 'inAutomaticMode' is true
Expected Completion	When the system is switched to Automatic mode
Risks and Assumptions	none
Responsibility	Train Controller

Table 48: System is in Normal Mode

Task	Test Design
Integrity Level	3
Methods	Compares 'inNormalMode' and 'inTestingMode'
Inputs	'inTestingMode' and 'inNormalMode', which are booleans
Outputs	'inTestingMode' is false and 'inNormalMode' is true
Expected Completion	When the system is switched to Normal mode
Risks and Assumptions	none
Responsibility	Train Controller

Table 49: System is in Testing Mode

Task	Test Design
Integrity Level	3
Methods	Compares 'inTestingMode' and 'inNormalMode'
Inputs	'inTestingMode' and 'inNormalMode', which are booleans
Outputs	'inTestingMode' is true and 'inNormalMode' is false
Expected Completion	When the system is switched to Testing mode
Risks and Assumptions	none
Responsibility	Train Controller

Table 50: Set Speed is Not Greater than Block Speed

Task	Test Design
Integrity Level	1
Methods	Compares the set speed and the block speed
Inputs	The block speed and the set speed
Outputs	The set speed is equal to the block speed
Expected Completion	When the 'Set Speed' button is clicked
Risks and Assumptions	The system is in Manual mode
Responsibility	Train Controller

Table 51: Set Speed is Not Greater than Suggested Speed

Task	Test Design
Integrity Level	1
Methods	Compares the set speed and the suggested speed
Inputs	The suggested speed and the set speed
Outputs	The set speed is equal to the suggested speed
Expected Completion	When the train needs to change speeds
Risks and Assumptions	The system is in Automatic mode
Responsibility	Train Controller

Table 52: Slider's Max Value is Equal to the Suggested Speed

Task	Test Design
Integrity Level	1
Methods	Compares the slider's max value to the suggested speed
Inputs	The suggested speed
Outputs	The suggested speed equals the max value of the slider
Expected Completion	When the suggested speed is changed
Risks and Assumptions	The system is in Automatic mode
Responsibility	Train Controller

Table 53: Slider?s Max Value is Equal to the Block Speed

Task	Test Design
Integrity Level	1
Methods	Compares the slider's max value to the block speed
Inputs	The block speed
Outputs	The block speed equals the max value
Expected Completion	When the block speed changes
Risks and Assumptions	The system is in Manual mode
Responsibility	Train Controller

9.2 Integration Tests

Table 54: Selecting a Train

Task	Test Design
Integrity Level	1
Methods	Compare the IDs of the two trains
Inputs	ID of the train selected
Outputs	The two IDs match
Expected Completion	When a train is selected from the dropdown menu
Risks and Assumptions	there is at least one dispatched train
Responsibility	Train Controller

Table 55: Turning AC On - Using Radio Button

Task	Test Design
Integrity Level	3
Methods	Compares the states of the AC and heat on the train
Inputs	The selected train
Outputs	The AC is on and the heat is off
Expected Completion	When the 'ON' radio button is selected
Risks and Assumptions	System is in Automatic or Manual mode, heat was on
Responsibility	Train Controller

Table 56: Turning AC On - Clicking Set

Task	Test Design
Integrity Level	3
Methods	Compares the states of the AC and heat on the train
Inputs	The selected train
Outputs	The AC is on and the heat is off
Expected Completion	When the 'Set' button is clicked
Risks and Assumptions	System is in Manual Mode, heat was on
Responsibility	Train Controller

Table 57: Turning Heat On - Using Radio Buttons

Task	Test Design
Integrity Level	3
Methods	Compares the states of the AC and the heat on the train
Inputs	The selected train
Outputs	The heat is on and the AC is off
Expected Completion	When the 'ON' radio button is selected
Risks and Assumptions	System is in Automatic or Manual mode, AC was on
Responsibility	Train Controller

Table 58: Turning Heat On - Clicking Set

Task	Test Design
Integrity Level	3
Methods	Compares the states of the AC and the heat on the train
Inputs	The selected train
Outputs	The heat is on and the AC is off
Expected Completion	When the 'Set' button is clicked
Risks and Assumptions	System is in Manual mode, AC was on
Responsibility	Train Controller

Table 59: Failures Window Reflects Failure on Train

Task	Test Design
Integrity Level	1
Methods	Compares the state of the train to the radio buttons
Inputs	The selected train
Outputs	The radio buttons match the failure on the train
Expected Completion	When a failure on the train occurs
Risks and Assumptions	the test checks all three (antenna, power, and brake) failures
Responsibility	Train Controller

Table 60: Sub-Component Receives Correct Train

Task	Test Design
Integrity Level	1
Methods	Compares the two IDs of the trains
Inputs	The selected train from the Train Cont.
Outputs	The two IDs match
Expected Completion	When a train is selected from the dropdown
Risks and Assumptions	none
Responsibility	Train Controller

10 Changelog

Table 61: Change

Date	Change
March 14, 2017	General Test Plan
March 15, 2017	Add more tests