**

Test Plan - Traffic Controller

Version 1.0

March 22, 2012

PAAC Demonstration System

|  |  |  |
| --- | --- | --- |
| List Of Revisions | | |
| Date | Name | Description |
| 3/22/2012 | Zhipeng Liu | Initial test plan document |

# Table of Contents

[List Of Revisions 2](#_Toc320186290)

[Table of Contents 3](#_Toc320186291)

[1 Introduction 4](#_Toc320186292)

[1.1 Purpose 4](#_Toc320186293)

[1.2 Scope 4](#_Toc320186294)

[1.3 Reference 4](#_Toc320186295)

[1.4 Definitions and Abbreviations 4](#_Toc320186296)

[2 Test Items 4](#_Toc320186297)

[3 Features to be Tested 4](#_Toc320186298)

[4 Testing Strategy 5](#_Toc320186300)

[4.1 Unit Testing 5](#_Toc320186301)

[4.2 Module Testing 5](#_Toc320186302)

[4.3 Item Pass/Fail Criteria 6](#_Toc320186303)

[4.4 Test Deliverables 6](#_Toc320186304)

[4.5 Responsibilities 6](#_Toc320186305)

# Introduction

## Purpose

The purpose of this document is to define the testing plan for the of the track controller module of the PAAC transit control prototype system.

## Scope

This document describes the testing approach and specific scenarios that will be tested in the track controller module.

## Reference

1. IEEE-830 Software Requirements Specification
2. NSC-009 PAAC Bid Package
3. Centralized Traffic Control Graphical User Interface Software Requirements Specification

## Definitions and Abbreviations

**Authority** – how far in distance the train is permitted to travel.

**Block** – a section of a railway line

**CTC** – Centralized Traffic Control

**GUI** – graphical user interface

**Region** **–** All the tracks between two switches

# Test Items

This document covers only the testing plan for the track controller independent of other modules in the system, with the exception of data contracts from the other modules needed for testing purposes. The test plan is intended for deliverable software candidates for the final product release and does not necessarily apply to interim software versions.

# Features to be Tested

# The test plan will target the following key features of the track controller module:

|  |  |
| --- | --- |
| Key features | Reference to the SRS |
| Information input from CTC Office | 3.1.1.1 |
| Information output to CTC Office | 3.1.1.2 |
| Information retrieving from Track Model | 3.1.1.3 |
| Information retrieving from track | 3.1.1.4 |
| Communication to track | 3.1.1.5 |
| Communication to other wayside controllers | 3.1.1.6 |
| Switches controlling | 3.1.4.1 |
| Railway crossings controlling | 3.1.4.2 |
| Failure detection of track blocks | 3.1.4.3 |

# Testing Strategy

## Overall Strategy

The testing for the track controller module will be split into two primary categories: Unit testing and Module Testing. The test cases should be reviewed for completeness prior to execution by a software developer other than the software author. The tests should also be run and composed into a report by a software developer other than the software author.

## Unit Testing

The track controller module will be unit tested using Microsoft Visual Studio 2010. The track controller module author will write corresponding unit tests to cover, at a minimum, each public method of every class in the software package. Each public method will be tested with various inputs to cover feasible scenarios (e.g. good data, invalid data, null data). Similar test cases will also be written to cover functional paths, i.e. methods that call other methods. The unit tests will be compiled into a program which can be executed with Visual Studio 2010. The results will be recorded in a deliverable test report.

## Module Testing

Test cases will be written to exercise the possible user interactions with the track controller module. The test cases will be targeted at verifying expected GUI responses and the messages sent to the transit system. Since the testing is intended to only test the track controller module, transit system interfaces will be replaced with stubbed classes to mimic system responses. The stubbed classes will record the messages passed to and from it in a log file. The log file will be compared with the expected output to verify the track controller module calls the interfaces correctly. Primary scenarios to be tested include the following:

|  |  |
| --- | --- |
| Key features | Reference to the SRS |
| CTC Office retrieves track information | 3.1.1.2, 3.1.1.4 |
| CTC Office sets train speed limit | 3.1.1.5.1 |
| CTC Office sets train authority | 3.1.1.5.2 |
| CTC Office sets switches | 3.1.1.5.4 |
| CTC Office sets signal lights | 3.1.1.5.5 |
| Trains track competition prevention | 3.1.3 |
| Trains emergency situation processing | 3.1.3 |
| Track failure processing | 3.1.4.3 |

## Item Pass/Fail Criteria

All unit tests and module tests are expected to pass. In the event that one or more tests fail, an assessment will be made by the development team to determine the severity of the error(s). The team will then decide on a course of action to resolve or mitigate the error(s), or to release the software with the defect(s) present.

## Test Deliverables

The following documents are to be delivered with the final software release:

* Unit Test Output and Report
* Module Test Cases
* Module Test Report

## Responsibilities

The software author is responsible for writing the unit test cases for each testable component of the software. The author is also responsible for writing the module test cases document. A developer other than the author is responsible for verifying the completeness and correctness of the unit tests and module test document. The same developer or another developer is responsible for executing the unit test program and module test and composing a test report for each. The software development team is collectively responsible for deciding on actions to be taken in the event that one or more test cases fail. The software author is then responsible for taking the decided action to resolve or mitigate the errors.