Test Plan

March 21, 2015

1 Introduction

This test plan has been created to check if our system conforms to all functional requirements and also to specify the types of testing which will be used to validate the functional requirements, action required on failed tests and time line to perform testing of the system.

2 Objectives and Tasks

2.1 Objectives

The objective of testing our system is to provide adequate testing of functional requirements, validation and behaviour of system under simulated condition. Exhaustive testing of system is not possible but we will use a diverse range of tests to find bugs and errors in the system including unit, functional, error and system testing. System is implemented using java therefore cross platform testing will be performed too. Some automation testing will be used with manual testing to find bugs are errors in the system.

2.2 Tasks

- Identifying functional requirements and writing the tests cases.
- Executing tests.
- Record the failed test cases and reporting them to team.
- Performing the re-test on failed test cases one bugs are fixed.

3 Scope

In context to our system, which is implemented using agent-based model it will be tested at three different levels of functionality.

- Agent behaviour:
  Can be checked by changing environment variables around agents. This forms functional requirements of the system as well. We have two agents in our system.

  1. Vehicles
     Following function requirements will be tested in Vehicle class
     - Accelerate
     - Move towards
     - Get vehicle ahead
     - Find best route
     - Slow down
  2. Traffic Lights
     - **Test Case 1:** Traffic Light Green
       Expected Output: Cars should keep moving if traffic light is green.
       Actual Output: Cars did not stop.
       Result: Passed.
     - **Test Case 2:** Traffic Light Red
       Expected Output: Cars should stop if traffic light is red.
       Actual Output: Cars stops when traffic light is red.
       Result: Passed
     - **Test Case 3:** Traffic Lights On More Than 2 Roads Junction
       Traffic lights should only be displayed on junctions that contain more than two roads.
       Expected Output: Only junctions with more than two roads display traffic lights.
       Actual Output: Traffic lights are displayed only on more than two road junctions.
       Result: Passed

- Runtime Parameters:
  Elements and data which loads at run time will form part of runtime testing. It will be tested, giving all valid and invalid inputs and check results. Following elements will be tested at run time:

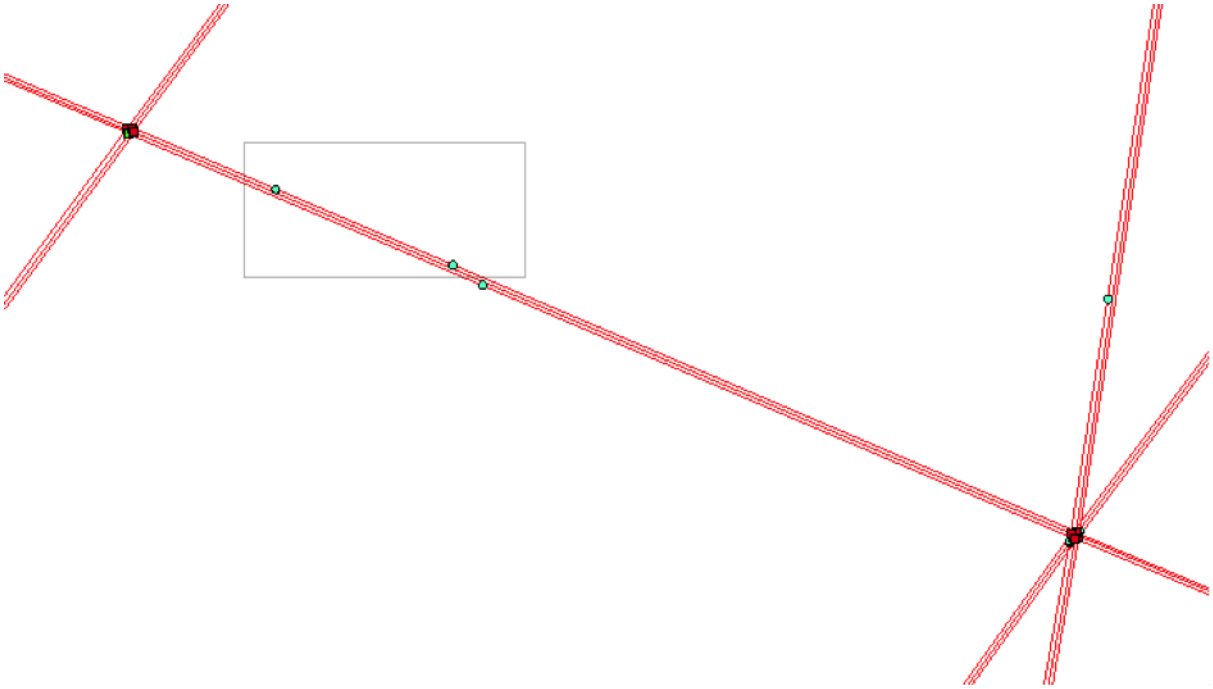
  - **Test Case 1:** SignalGreenBuilder Implements ContextBuilder
    This class contains methods to build the context and initialise all variables required during the simulation execution.
    Input: Run signalGreen Model, to load user GUI which contains GIS map, Vehicle agents and TrafficLight Agents. Expected Output: Map should be loaded and displayed along with three different types of vehicles and traffic

lights on all junctions that have more than two edges. Actual Output: Map loaded with 100 vehicles which is default value and traffic lights on junctions with more than two edges only. Result: Passed

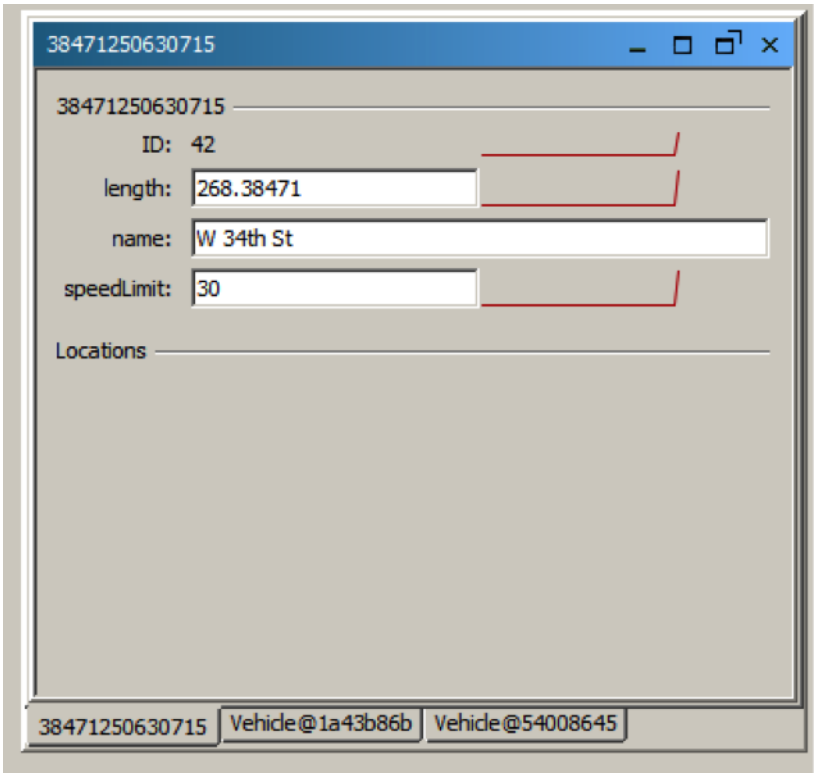
- **Test Case 2:** Number of vehicles
  Input: Enter a number in "Number of vehicles" field on parameters tab.
  4, 0 , -1 and a character given as input.
  Expected Output: Only entered number of vehicles should appear on road network. No vehicles should appear for negative numbers and characters.
  Actual Output: Vehicles are equal to number entered. No vehicles for negative and character input.

  Result: Passed
- **Test Case 3:** No Traffic Lights
  "Traffic Lights" on parameters tab, if unchecked traffic lights should not appear on display and cars should not consider traffic lights.
  Expected Output: Cars should travel on road without stopping at traffic lights as there are none.
  Actual Output: No traffic lights and cars carry on moving on roads.

- Overall behaviour of system:
  Checks that all agents of the system produce expected results in a given scenario. This will include testing of agents interaction with other components of environment (roads, traffic lights) when programme will be executed at run time.
  Test Case 1: Every junction holds a queue of vehicles running on a particular outward road.
  Expected Output of next.printVehiclesQueue(origin):[signalGreen.Vehicle@54008645, signalGreen.Vehicle@1a43b86b] Peek vehicle: signalGreen.Vehicle@54008645
  Actual Output: Using probe tool, selecting area which we want to inspect:



There are indeed two vehicles running on W 34th St, and their object ID match.



Result: Passed