NETWORK FLOW APPLICATION

TEST PLAN

Team name
Zoe Fu
Reid Stagemeyer
Dat Nguyen

I. INTRODUCTION

Describe high level test plan objectives, such as features to be tested and type of testing. The goal is to provide a framework that can be used by managers and tester to plan and execute the necessary test in a timely and cost-effective manner.

II. SYSTEM OVERVIEW

The proposed system incorporates the Model, View, Controller design pattern. The User (Mayor Mann) interacts through the console (Controller) while the Map object with its individual traffic components (Model) contains the relevant data. The data and its resulting statistics from the model can be viewed via the Report class (View).

The proposed system is broken down into 4 parts: Interface, Display, Component, and Simulator. The architecture model is shown graphically in figure III.a.

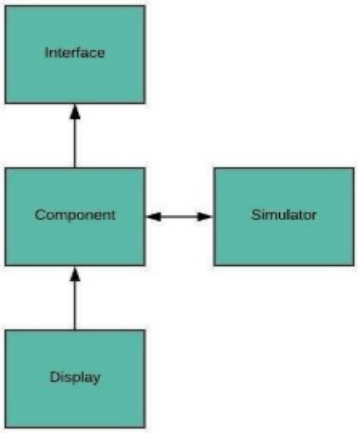

Figure III. a show subsystem of networkflow app

III. FEATURES TO BE TESTED

Testing Requirement	System Requirement(s)	Short Description	Testing Status
1	F1, N1b	Execute the app and a display pop up.	Manually
2	F2, N3b, N4b	Input different map layouts created from web app. Make sure that the created map in the app and its objects match the imported map.	Junit
3	F2, N3b, N4b	Input a non-json file type	Junit
4	F3, N5a	Place a car near a traffic light and stop sign then observe the car state and traffic light state in UserView window	Manually
5	F4, N1a, N3a	Visualized car information matches with car object information. Create a map with straight road then output the	Junit

Commented [dn1]: Ask brian

Commented [dn2R1]:

Commented [dn3]: Add global timer. Print out car state at each time step.

Commented [dn4]: Print out car list and compare with input json.

		index of the car on map in every step until it reaches destination.	
6	F4, N1a, N3a	Place a car in front of a stop sign then output the next tile depending on car's chosen turning direction.	Junit
7	N2b	Add Traffic light and stop sign on a map without road around.	Junit
8	N4a	Run the application on multiple platform and output OS type.	Manually
9	N3a, N3b	Test system to handle multiple cars and components in the map. Input map with multiple cars and components	Junit
10	N2a	Attempt to place multiple car on map and observe the behaviors	Manually
11	F4	Place 4 car in different directions at a stop sign and observe the behaviors	Manually

Commented [dn5]: Check if there are no built in direction then print warning.

Commented [dn6]: Print out os info when running the apps

IV. TESTING ENVIRONMENT

Hardware: Linux, Window, and OS computer.

Software: Eclipse

Junit java testing framework will be used to test Java Implementation. Javascript implementation will be tested automatic with a bash script and input/expected folders.

TEST CASES

TEST CASE 1

COMPONENT UNDER TEST

AppMain – view subsystem

FEATURE(S) TO BE TESTED

F1, N1b

INTIAL CONDITIONS

Run system source code with Eclipse IDE

EXPECTED BEHAVIOR

INPUT

Click start in Eclipse or run the apps by executing the jar files

OUTPUT

Display pop up showing the simulation map and 2 user controllable windows.

TEST CASE 2

COMPONENT UNDER TEST

SimulationMap – Model subsystem

FEATURE(S) TO BE TESTED

F2, N3b, N4b

INTIAL CONDITIONS

Run system source code with Eclipse IDE

EXPECTED BEHAVIOR

INPUT

Path to a json file with Following content

{"numHeight":"3","numWidth":"3","tiles":[[{"generalType":"ground","classType":"ground"},{"generalType":"ground","classType":"ground"},{"id":1,"generalType":"traffic-light","classType":"traffic-light","builtDirections":"","xIndex":2,"yIndex":0}],[{"id":3,"generalType":"stop-sign","classType":"stop-sign","builtDirections":"","xIndex":0,"yIndex":1},{"generalType":"ground","classType":"ground"},{"generalType":"ground","classType":"ground"}],[{"generalType":"road","classType":"road-horizontal"},{"generalType":"road","classType":"road-horizontal"},{"generalType":"road","classType":"road-horizontal"}]],"trafficComponents":[{"id":1,"generalType":"traffic-light","classType":"traffic-

light","builtDirections":"","xIndex":2,"yIndex":0},{"id":3,"generalType":"stop-sign","classType":"stop-sign","builtDirections":"","xIndex":0,"yIndex":1}],"cars":[]}

OUTPUT

'ground, ground, traffic-light,

Stop-sign, ground, ground,

Road, road, road'

TEST CASE 3

COMPONENT UNDER TEST

SimulationMap – Model subsystem

FEATURE(S) TO BE TESTED

F2, N3b, N4b

INTIAL CONDITIONS

Prepare a Non json file and Run system source code with Eclipse IDE

EXPECTED BEHAVIOR

INPUT

Path to non-json extension file

OUTPUT

Raise Error saying "Non json file extension"

TEST CASE 4

COMPONENT UNDER TEST

Car – Model subsystem

FEATURE(S) TO BE TESTED

F3, N5a

INTIAL CONDITIONS

A Json file exported from createmap Web App. Run system source code with Eclipse IDE

EXPECTED BEHAVIOR

INPUT

Create a map, place a car near traffic light and output the sequence of state over times untils the car completes the turn

OUTPUT

State transition: Idle -> stop -> turn -> regular. Car correctly at the correct location base on observation

TEST CASE 5

COMPONENT UNDER TEST

Car, Road – Model Subsystem

FEATURE(S) TO BE TESTED

F4, N1a, N3a

INTIAL CONDITIONS

Run system source code with Eclipse IDE

EXPECTED BEHAVIOR

INPUT

Car path to json file where a car is placed on a straight road and output the car next state.

OUTPUT

True(meaning movable)

TEST CASE 6

COMPONENT UNDER TEST

Car, Road – Model Subsystem

FEATURE(S) TO BE TESTED

F4, N1a, N3a

INTIAL CONDITIONS

Run system source code with Eclipse IDE

EXPECTED BEHAVIOR

INPUT

Input path to json file holding map information where a car is placed in front of a stop sign

OUTPUT

False(Not movable)

TEST CASE 7

COMPONENT UNDER TEST

SimulationMap, StopSign, TrafficLight – Model Subsystem

FEATURE(S) TO BE TESTED

N2b

INTIAL CONDITIONS

Run system source code with Eclipse IDE

EXPECTED BEHAVIOR

INPUT

Path to json file where stop sign and traffic light does not have road in all directions. Contents of the json files are shown as below.

{"numHeight":"4","numWidth":"4","tiles":[[{"generalType":"ground","classType":"grass"},{"generalType":"ground","classType":"grass"},{"generalType":"ground","classType":"grass"},{"id":8,"generalType":"stop-sign","classType":"stop-sign","builtDirections":"","xIndex":3,"yIndex":0}],[{"generalType":"ground","classType":"grass"},{"id":4,"generalType":"stop-sign","classType":"stop-sign","builtDirections":"","xIndex":1,"yIndex":1},{"generalType":"ground","classType":"grass"},{"generalType":"ground","classType":"grass"}],[{"generalType":"ground","classType":"grass"},{"generalType":"ground","classType":"grass"},{"generalType":"ground","classType":"grass"},{"generalType":"ground","classType":"grass"}],[{"id":7,"generalType":"stop-sign","classType":"stop-sign","builtDirections":"","xIndex":0,"yIndex":3},{"generalType":"ground","classType":"grass"},{"generalType":"ground","classType":"grass"},{"id":6,"generalType":"traffic-light","classType":"traffic-light","builtDirections":"","xIndex":3,"yIndex":3}]],"trafficComponents":[{"id":8,"generalType":"stop-sign","classType":"stop-sign","builtDirections":"","xIndex":3,"yIndex":0},{"id":4,"generalType":"stop-sign","classType":"stop-sign","builtDirections":"","xIndex":1,"yIndex":1},{"id":7,"generalType":"stop-sign","classType":"stop-sign","builtDirections":"","xIndex":0,"yIndex":3},{"id":6,"generalType":"traffic-light","classType":"traffic-light","builtDirections":"","xIndex":3,"yIndex":3}],"cars":[]}

OUTPUT

Warning: Intersection with out road in all 4 directions.

TEST CASE 8

COMPONENT UNDER TEST

AppMain – view subsystem

FEATURE(S) TO BE TESTED

N4a

INTIAL CONDITIONS

Run system source code with Eclipse IDE

EXPECTED BEHAVIOR

INPUT

Run the program on Window machine. Simulate the apps then print the System its running on

Thanks to: https://stackoverflow.com/questions/228477/how-do-i-programmatically-determine-operating-system-in-java

Run the program on Mac machine. Simulate the apps then print the System its running on

OUTPUT

Window

Mac

TEST CASE 9

COMPONENT UNDER TEST

SimulationMap, Car - Model

FEATURE(S) TO BE TESTED

N3a, N3b

INTIAL CONDITIONS

Run system source code with Eclipse IDE

EXPECTED BEHAVIOR

INPUT

Path to json file holding information of multiple cars and traffic components on map. Content is shown as follow.

{"numHeight":"6","numWidth":"6","tiles":[[{"id":8,"generalType":"traffic-light","classType":"traffic-light","builtDirections":"","xIndex":0,"yIndex":0},{"generalType":"ground","classType":"grass"},{"generalType":"ground","classType":"grass"},{"generalType":"ground","classType":"grass"},{"generalType":"ground","classType":"grass"},{"generalType":"road","classType":"road-horizontal"}],[{"generalType":"ground","classType":"grass"},{"id":3,"generalType":"stop-sign","classType":"stop-sign","builtDirections":"","xIndex":1,"yIndex":1},{"generalType":"ground","classType":"grass"},{"generalType":"ground","classType":"grass"},{"id":5,"generalType":"traffic-light","classType":"traffic-light","builtDirections":"","xIndex":4,"yIndex":1},{"generalType":"ground","classType":"grass"}],[{"generalType":"ground","classType":"grass"},{"generalType":"ground","classType":"grass"},{"generalType":"ground","classType":"grass"},{"generalType":"ground","classType":"grass"},{"generalType":"ground","classType":"grass"},{"generalType":"ground","classType":"grass"}],[{"generalType":"ground","classType":"grass"},{"generalType":"ground","classType":"grass"},{"id":4,"generalType":"stop-sign","classType":"stop-sign","builtDirections":"","xIndex":2,"yIndex":3},{"generalType":"ground","classType":"grass"},{"generalType":"ground","classType":"grass"},{"generalType":"ground","classType":"grass"}],[{"generalType":"road","classType":"road-horizontal"},{"generalType":"ground","classType":"grass"},{"generalType":"ground","classType":"grass"},{"generalType":"ground","classType":"grass"},{"generalType":"ground","classType":"grass"},{"generalType":"ground","classType":"grass"}],[{"generalType":"ground","classType":"grass"},{"generalType":"ground","classType":"grass"},{"generalType":"ground","classType":"grass"},{"generalType":"ground","classType":"grass"},{"generalType":"ground","classType":"grass"},{"generalType":"road","classType":"road-horizontal"}]],"trafficComponents":[{"id":8,"generalType":"traffic-light","classType":"traffic-light","builtDirections":"","xIndex":0,"yIndex":0},{"id":3,"generalType":"stop-sign","classType":"stop-sign","builtDirections":"","xIndex":1,"yIndex":1},{"id":5,"generalType":"traffic-light","classType":"traffic-

light","builtDirections":"","xIndex":4,"yIndex":1},{"id":4,"generalType":"stop-sign","classType":"stop-sign","builtDirections":"","xIndex":2,"yIndex":3}],"cars":[{"xIndex":1,"yIndex":2,"pixi.position.x":75,"pixi.position.y":125,"direction":">","tick":0,"state":"idle"},{"xIndex":0,"yIndex":4,"pixi.position.x":25,"pixi.position.y":225,"direction":">","tick":0,"state":"idle"},{"xIndex":5,"yIndex":5,"pixi.position.x":275,"pixi.position.y":275,"direction":"<","tick":0,"state":"idle"},{"xIndex":5,"yIndex":0,"pixi.position.x":275,"pixi.position.y":25,"direction":"<","tick":0,"state":"idle"}]}

OUTPUT

4 traffic components and 4 cars. Validate the correct components

TEST CASE 10

COMPONENT UNDER TEST

AppMain, Car, and SimulationMap - Model

FEATURE(S) TO BE TESTED

N3a

INTIAL CONDITIONS

Run system source code with Eclipse IDE

EXPECTED BEHAVIOR

INPUT

Attempt to place multiple cars on map

OUTPUT

Car visually turns at intersection and continue moving if next tile is road.

TEST CASE 11

COMPONENT UNDER TEST

AppMain, Car, and SimulationMap – Model and view subsystem

FEATURE(S) TO BE TESTED

F4

INTIAL CONDITIONS

Run system source code with Eclipse IDE. Json file

EXPECTED BEHAVIOR

INPUT

Input path to json file where 4 car is placed in different direction at a stop sign

OUTPUT

Cars successfully stop and move in stopsign