*Florida International University*

*School of Computing and Information Sciences*

CIS 4911 - Senior Capstone Project

Software Engineering Focus

Final Deliverable

**Informed Traveler**

**or**

**University City Traffic Simulation**

**Ver-1.0**

Team #5

**Team Members**

Artiom Tiurin

Daniel Costa e Sa

**Product Owners**: Gerald Inberg, Oliver Ullrich, Charaf Azzouzi

**Instructor**: Masoud Sadjadi

Copyright and trademark notices, restrictions on copying or distributing the documentation, information for contacting the issuing organization (reader’s comments), warranties, contractual obligations or disclaimers, and general warnings and cautions.

***Abstract***

Traffic in the city of Sweetwater is just as much of a problem as it is for any city, maybe even more since the city of Sweetwater is in the center of major streets and concentrated between Turnpike and SR-836 highways used by all the surrounding cities. The UniversityCity Sweetwater Traffic Simulation(USTS) is a model that utilizes agent based simulation for the purposes of recreating the current traffic conditions that are experienced by the city Sweetwater and attempt to find timing optimizations that the streetlights could employ to reduce traffic. Using the original implementation of the traffic simulation for the Barbarossaplatz region in the city of Cologne, the USTS uses the objects from that research project to build an underlying model. Further, data was needed to correctly simulate: lane lengths, transportation infrastructure layout, and car-agent behavior.

**Table of Contents**

Introduction............................................................................................................................. 5

Current System ...................................................................................................................... 5

Purpose of New System ......................................................................................................... 5

User Stories ........................................................................................................................... 6

Project Plan............................................................................................................................ 7

Hardware and Software Resources ........................................................................................7

Sprints Plan ........................................................................................................................... 8

Sprint 1 .................................................................................................................................. 8

Sprint 2 .................................................................................................................................. 9

Sprint 3 .................................................................................................................................. 12

Sprint 4 .................................................................................................................................. 17

Sprint 5 .................................................................................................................................. 18

System Design........................................................................................................................ 23

Architectural Patterns ............................................................................................................. 24

System and Subsystem Decomposition ...................................................................................25

Deployment Diagram .............................................................................................................. 30

Design Patterns ...................................................................................................................... 30

System Validation ................................................................................................................... 31

Appendix A - UML Diagrams ................................................................................................... 33

Static UML Diagrams .............................................................................................................. 33

Dynamic UML Diagrams ......................................................................................................... 43

Appendix B - User Interface Design ........................................................................................ 47

Appendix C - Sprint Review Reports ...................................................................................... .50

Appendix D - Sprint Retrospective Reports ............................................................................. 52

References .............................................................................................................................. 53

# 

# 

# **Introduction**

This document describes in detail the processes, artifacts generated within this project. The document is covered in order of the work done starting with description of the current system, requirements elicitation, product development cycle, system design, implementation, and verification.

The entire process was performed in a disciplined, quantifiable, and iterative manner starting with the source code study and requirements elicitation in May of 2015, followed by user interface design and development in June 2015, and finally iterative cycles of implementation and verification in July of 2015.

## Current System

The current system consists of packages that contain objects used to create a model of a road system, and then logic used to simulate traffic using agents that make use of the model. The model found in the current system was one that reflected the region of Barbarossaplatz in Cologne, Germany. Researchers in that region were attempting to find ways to optimize the traffic of cars in the streets by manipulating the timings of street lights. In order to accomplish this they first built the model that accurately reflected the same conditions found in Barbarossaplatz. The model they built was flexible and customizable enough where different conditions could be simulated in such a way that hypothetical conditions such as emergencies could also be ran in the model.

## Purpose of New System

The new system shall use the same objects found in the current system to build a model of the city of Sweetwater so that researchers can simulate both accurate and hypothetical traffic conditions. The new system shall have the same goals as the current system, which is using agent based simulation to find ways to optimize the traffic of cars in the streets by manipulating the timings of street lights.

**User Stories**

For the proposed system, first it was required to have a solid foundation of understanding of the model of the Barbarossaplatz region and all of the objects used to create the model, how those objects interacted with each other, and how the system manipulates those objects to reflect real world conditions. Then a research was needed to gather information about the road infrastructure of the city of Sweetwater so that a model could be created to reflect it. Once that information was gathered then work on creating the objects and putting them together was undertaken. Once the model was finished, and extensively tested, more information was needed to ensure that the agents behavior reflected that of the real world, as well as certain objects within the model such as street light timings and average car capacity of each lane throughout different times of the day. The next step was to update the gui so that the statistics given in the panels were those of the new model and not of the old one, after which some additional functionality was added to gui such as being able to export the model and visualize on screen statistics about the agents.

For a comprehensive list of all the user stories please see the section Sprint Plan.

Refer to Use Case Diagram in Appendix A.

# 

# 

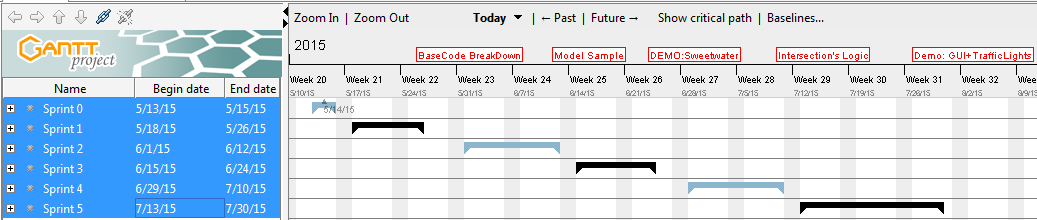
# **Project Plan**

The plan is for the project to provide comprehensive real-time and predictive decision support in the areas of multi-modal and inter-modal transportation.

It will optimize local decision making by providing individual customers with custom-tailored information on available multi-modal and inter-modal transportation options, and optimize global traffic management by providing transportation service providers with management support to improve efficiency and effectiveness.

Daniel Costa e Sa - Cross functional and software engineer

Artiom Tiurin - Cross functional and software engineer



Hardware and Software Resources

**Hardware**

The following are the specification of the computer that will be used for the development of this project:

**Laptop:**

**- Processing Power:** 2.1 GHz Intel Core i7

**- RAM Memory:** 16 GB 1600 MHz DDR3

**- Available space on hard drive:** 167 GB flash storage

**Other Devices**

- **Input devices**

-Keyboard- Standard QWERTY keyboard

-Mouse - Trackpad mouse

- **Output devices**

-Monitor display- 15.1-inch (1366 x 786) Intel HD Graphics 4000 graphics

**Software**

1. **Windows 7 -** operating system used to run tools for development
2. **Netbeans 8.0.2** - IDE tool, used to develop code and some diagrams for the project.
3. **Visio 2013 -** tool used to build various graphs and UML diagrams
4. **Google Docs -** used to updated and keep documentation
5. **Google Maps -**  used for object reference
6. **Java Swing Utility -** a java library used for visualization and GUI.

## 

## 

## 

## Sprints Plan

### Sprint 1

(5/18/2015 - 5/29/2015)

# **User Story # 114 - Create class diagrams for existing code base**

***Tasks***

* Research tools for creating UML diagrams of existing code base
* Test each tool
* Get familiarized with the project

***Acceptance Criteria***

* Class diagram that shows dependencies b/w classes
* Project’s structure is visible

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

### 

### 

### 

### Sprint 2

(6/1/2015 - 6/12/2015)

# **User Story # 126 - Create Lane Graph**

***Tasks***

* Gather lengths of all lanes to be modeled.
* Place Lanes with their length onto a scale and gather coordinates.
* Place all info into excel and a graph.

***Acceptance Criteria***

* Visual Lane graph of Sweetwater perimeter.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

# **User Story # 127 - Create Traffic Light Circuit Graph**

***Tasks***

* Complete Traffic Light Circuit Graph, so it will be easier to translate model into code.

***Acceptance Criteria***

* Traffic light Circuit diagram that displays all Sweetwater major roads.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

# **User Story # 128 - Implement the Traffic Lane model for 8th Str/112th Ave**

***Tasks***

* Use the information provided on the Lane graph to hard code the objects necessary for implementing the specified area of the model.

***Acceptance Criteria***

* Visual representation of the model in GUI.
* Model entities are visible in GUI.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

# **User Story # 132 - Implement Traffic Lane Graph for 8th Str/109th Ave**

***Tasks***

* Use the information provided on the Lane graph to hard code the objects necessary for implementing the specified area of the model.

***Acceptance Criteria***

* Visual representation of the model in GUI.
* Model entities are visible in GUI.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

# **User Story # 133 - Implement Traffic Lane Graph of intersection 8St/107Ave**

***Tasks***

* Use the information provided on the Lane graph to hard code the objects necessary for implementing the specified area of the model.

***Acceptance Criteria***

* Visual representation of the model in GUI.
* Model entities are visible in GUI.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

### 

### Sprint 3

(6/15/2015 - 6/26/2015)

# **User Story # 129 - Integrate Traffic Lane model(8th Str/112th Ave) into the Base Code**

***Tasks***

* Test and correct errors that were in the hard code of the objects necessary for implementing the specified area of the model.

***Acceptance Criteria***

* The project is runnable.
* Agents use the objects properly.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

# **User Story # 90 - Integrate Traffic Lane model(8th Str/109th Ave) into Base Code**

***Tasks***

* Test and correct errors that were in the hard code of the objects necessary for implementing the specified area of the model.

***Acceptance Criteria***

* The project is runnable.
* Agents use the objects properly.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

# **User Story # 134 - Integrate Traffic Lane model(8th Str/107th Ave) into Base Code**

***Tasks***

* Test and correct errors that were in the hard code of the objects necessary for implementing the specified area of the model.

***Acceptance Criteria***

* The project is runnable.
* Agents use the objects properly.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

# **User Story # 153 - Implement Traffic Lane Graph of intersection Flagler and 107th**

***Tasks***

* Use the information provided on the Lane graph to hard code the objects necessary for implementing the specified area of the model.

***Acceptance Criteria***

* Visual representation of the model in GUI.
* Model entities are visible in GUI.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

# **User Story # 154 - Implement Traffic Lane Graph of intersection Flagler and 112th**

***Tasks***

* Use the information provided on the Lane graph to hard code the objects necessary for implementing the specified area of the model.

***Acceptance Criteria***

* Visual representation of the model in GUI.
* Model entities are visible in GUI.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

# **User Story # 155 - Implement Traffic Lane Graph of intersection Flagler and 109th**

***Tasks***

* Use the information provided on the Lane graph to hard code the objects necessary for implementing the specified area of the model.

***Acceptance Criteria***

* Visual representation of the model in GUI.
* Model entities are visible in GUI.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

# **User Story # 156 - Integrate Traffic Lane model(Flagler/107th Ave) into Base Code**

***Tasks***

* Test and correct errors that were in the hard code of the objects necessary for implementing the specified area of the model.

***Acceptance Criteria***

* The project is runnable.
* Agents use the objects properly.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

# **User Story # 157 - Integrate Traffic Lane model(Flagler/109th Ave) into Base Code**

***Tasks***

* Test and correct errors that were in the hard code of the objects necessary for implementing the specified area of the model.

***Acceptance Criteria***

* The project is runnable.
* Agents use the objects properly.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

# **User Story # 167 - Integrate Traffic Lane model(Flagler/112th Ave) into Base Code**

***Tasks***

* Test and correct errors that were in the hard code of the objects necessary for implementing the specified area of the model.

***Acceptance Criteria***

* The project is runnable.
* Agents use the objects properly.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

### Sprint 4

(6/29/2015 - 7/10/2015)

# **User Story # 98 - Implement connection points: I, J, and D**

# 

***Tasks***

* Implement connection points: Intersections, Junctions, Dissections, so they resemble real world events.

***Acceptance Criteria***

* I,J,D implemented as per logic of the system, based on Base Code.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

# **User Story # 174 - Integrate connection points: I, J, and D with Base Code**

# 

***Tasks***

* Test and correct errors that were in the hard code of the objects necessary for implementing the specified area of the model.

***Acceptance Criteria***

* The project is runnable.
* Agents use the objects properly.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

### Sprint 5

(7/13/2015 - 7/24/2015)

# **User Story # 190 - Export of represented model graph**

# 

***Tasks***

* Add the functionality of being able to export a visual representation of the universitycity traffic model in different image formats (png, jpeg, svg, etc..).

***Acceptance Criteria***

* User friendly graphic user interface for the export function
* User is able to export the model graph in different image formats
* User have the possibility to choose a name, a location and a type of the file to be saved

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

# **User Story # 188 - Add to documentation**

# 

***Tasks***

* Add to the growing documentation for the universitycity clear water traffic simulation.

***Acceptance Criteria***

* Each section of the documentation is completed.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

# **User Story # 189 - Show/hide street name at point and click**

# 

***Tasks***

* add the functionality of being able to Show/hide the street name by just pointing and clicking, after which it will be visible.

***Acceptance Criteria***

* Information about the agent is visible after clicking on it.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

# **User Story # 178 - Fix LiveStatsPanel.java**

# 

***Tasks***

* Make sure the objects displayed by LiveStatsPanel.java has been adjusted to the current model.

***Acceptance Criteria***

* Have stats bar show up properly.
* Have the real stats show up.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

# **User Story # 107 - Collect Traffic real world data**

# 

***Tasks***

* Gather real traffic data collected per lane, so the traffic conditions could be simulated based on real events.

***Acceptance Criteria***

* Data is entered in a DataSheets

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

# **User Story # 173 - Update Documentation**

# 

***Tasks***

* Add to the growing documentation for the universitycity clear water traffic simulation.

***Acceptance Criteria***

* User Stories correspond with use cases.
* Use Cases correspond with Seq. Diagrams.
* Seq. Diagrams correspond with Class Diagrams.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

# **User Story # 175 - Implement and Integrate Traffic Events**

# 

***Tasks***

* Hard code the data needed to run events based on collected real traffic data, so the project resembles the real world.

***Acceptance Criteria***

* Events timing based on collected data.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

# **User Story # 97 - Gather traffic signal data**

# 

***Tasks***

* Create a dataset of traffic signal timing for each intersection based off real world data, so the Traffic Light Circuit Graph could be put together.

***Acceptance Criteria***

* Data gathered from real world sources
* Traffic Light Circuit Graph is drawn in Visio and displays times for each intersection.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

# **User Story # 200 - Display information about a single agent**

# 

***Tasks***

* Display information e. g. speed, agent id, acceleration, etc… about a single agent (car) running in the simulator.

***Acceptance Criteria***

* By selecting/clicking one agent should display information about this agent to the user.
* The information to be displayed in pop-up window or in tooltip.
* It should be displayed at least three properties/information about the selected agent

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

# **User Story # 201 - Create additional Springs Strengths Modificators**

***Tasks***

* Implement Strength Modificators for all implemented Springs, so I will be able to manipulate amount of cars incoming for each Spring

***Acceptance Criteria***

* Strength Mods are visible in dropdown menu during the run of the project.
* Change of the Spring bar affects the Spring strength.

***Modeling***

Refer to UML diagrams in Appendix A that were created or modified to model the functionality that will be implemented in this sprint.

# **User Story # 205 - Create a Database for entities**

# 

***Tasks***

* Create an initial setup of Database for Entitiy component, so the next team will be using it.

***Acceptance Criteria***

* Database setup in SQL Server.
* Each entity is a table.

***Modeling***

Refer to UML diagrams in Appendix A : Figure D

# 

# 

# **System Design**

There is a Simulation package that contains the Main method needed to start the simulation. This package calls on the sweetwatercity package which contains the files that hold the objects for the actual Sweetwater city model. The simulation package also calls the gui package which contains all the components that give the end user the graphic user interface that is used to see not only the visualization of the model and the agents as they navigate the model, but the end user can also control the simulation using some of the controls found in the GUI. In order to give the system the ability to manipulate the objects the entities package contains information about all the objects used to create a model, how they are initialized, the parameters they have, and the methods used to manipulate them.

## Architectural Patterns

A component based architecture, coupled with a Model-View-Controller pattern, was chosen for the new system because this emphasizes theseparation of concerns in respect of the wide-ranging functionality available throughout a given system software. Building models and running simulations on them is easier to implement when using the Model-View-Controller pattern, which allows the developers to separately work on the data needed to build the model, how the system manipulates the data of the model, and how the system displays the model and the agents to the user. This allowed the developers to use a reuse-based approach to defining, implementing and composing loosely coupled independent components into systems. This allowed us to bring about an equally wide-ranging degree of benefits in both the short-term and the long-term for the software itself and for organizations that sponsor such software.

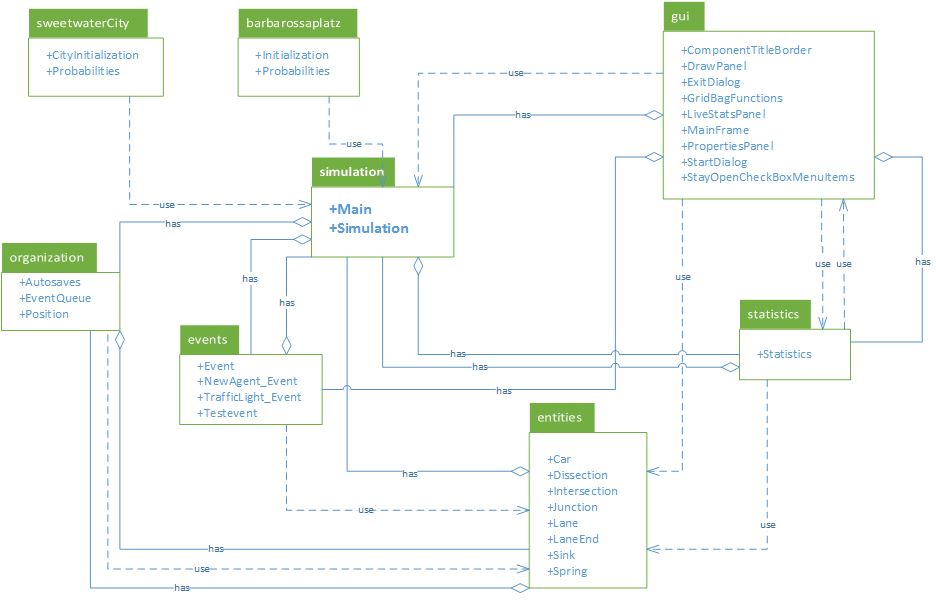
## 

## 

## 

## 

## System and Subsystem Decomposition



**Figure P: Package Diagram**

The current system is divided into the following components or packages:

* **sweetwaterCity**

Most of the model specific code is kept here, where objects and behavior specific to the sweetwater region are hardcoded.

* + InitSweetwater.java

All non-agent objects used to create the model are initialized here. First the all the lanes are initialized, beginning with Springs(labeled as A), then Sinks(labeled as V), regular lanes(labeled as M), and connector lanes(labeled as K). Lanes are the basic units used by the model to define streets and as of now the only thing differentiating all the lanes are their labels, more objects have to be created and added to the lane objects to obtain a specific functionality from each. After all the lanes are created, the actual Spring and Sink objects are added to the lane objects. Springs are the entrances to the model for the car agents and sinks are the exit points. Then Dissections, Intersections, and Junctions are added to the ends of the other lanes not being used as Sinks. Each of the three are objects that connect lanes to each other and also define lane switching behavior for that lane. Then endPoints are set, which make explicit the neighboring lanes of each lane. Then the initial state of each traffic light is set as an event, afterwards the events responsible for adding agents to the model are then added.

Please see the the respective files for each object in the entities package for more specific information and the necessary parameters for each.

* + probabilities.java

This file defines the likelihood of the new directions of the the agents as they exit one lane. Some lanes flow into multiple lanes, so the likelihood an agent chooses one lane over another comes from real world observations and research.

* **entities**

This package contains information about all the objects used to create the model, how they are initialized, the parameters they have, and the methods used to manipulate them.

* + car.java
  + dissection.java
    - Many lanes in, many lanes out, allows lane changes
  + intersection.java
    - One lane in, one lane out, no lane changes
  + junction.java
    - Many lanes in, many lanes out, does not allow lane changes
  + lane.java
  + laneend.java
  + sink.java
  + spring.java
* **events**
  + event.java
  + newagent\_event.java
  + testevent.java
  + trafficlight\_event.java
* **gui**

This package contains all the components that give the end user the graphic user interface that is used to see not only the visualization of the model and the agents as they navigate the model, but the end user can also control the simulation using some of the controls found in the GUI.

* + componenttitleborder.java
  + drawpanel.java
  + exitdialog.java
  + gridbagfunctions.java
  + livestatpanel.java
  + mainframe.java
  + propertiespanel.java
  + startdialog.java
  + stayopencheckboxmenuitem.java
* **gui.resources**

This package contains the .properties files used by the gui package in order to run. These properties are mostly strings that are used as labels for the GUI and get changed when the language setting is changed.

* + messagebundle-copy.properties
  + messagebundle.properties
  + messagebundle\_de\_de.properties
  + messagebundle\_en\_us.properties
  + messagebundle\_es\_es.properties
* **organization**

This package contains some files that provide some minor classes used to organize some of the data used by the model, this is so that the data is more manageable and the logic used to manipulate is also more readable from a coding point of view.

* + autosaves.java
  + eventqueue.java
  + position.java
* **simulation**

This file contains the Main.java file for intializing the program and the simulation.java which contains some global variables used by the whole model to control some aspects of the agents behavior within the model.

* + main.java
  + simulation.java
* **statistics**

This package contains a single file that contains methods that are meant to produce different results based on different “cases”. These cases are chosen according to probabilities that are set elsewhere.

* + Statistics.java
* **tests**

This package contains a set of test cases that was used when the base code was first created. These tests were mainly for seeing how well a model generated would perform, as well as the communication between the gui components and the model.

* + awttests.java
  + clipdemo.java
  + drawfirstline.java
  + drawstringdemo.java
  + paintindeirect.java
  + protosimframe.java
  + test.java
  + paintxorufo.java

## 

## 

## 

## 

## 

## Deployment

All subsystems are kept in a central program. The system contains functionality for models to be kept in a database but no models are kept there yet. As the system stands now the system does not include components for the web or a server. All parts are deployable from a desktop, including the models. Computational resources are not distributed throughout multiple machines but are ran from just a single one, which happens to be the one that originally ran the simulation.

## Design Patterns

The proposed system makes heavy use of the behavioral pattern known as the state pattern. the agents used by the simulation have to constantly check their surroundings for other agents(cars that share the road with them), this information is kept as a parameter inside the agent that gets updated by the system. If they are too close to another car or a car is stopped ahead of them then the agent needs to alter its behavior, by either stopping, slowing down, or switching lanes. Use of the state pattern to change the behavior once internal parameters are changed allow the system to make coordination of multiple number of agents feasible and more manageable, as well as easier for developers to test and maintain.

**System Validation**

In this section we used JUNIT build in testing tool for NetBeans, to test some of the functionalities of the Traffic Simulator.

**User Story # 130 - Create Test Cases**

Related User Stories # 131 - Implement Test Cases

**RAINY DAY:**

|  |  |
| --- | --- |
| Test ID | UCTS\_01\_Run Simulation |
| Purpose | To test the run of the simulation in initialization |
| Preconditions | User is able to open project in NetBeans |
| Inputs | null in place of the seed |
| Expected Response | JUNIT- FAIL:  GUI - Launch  Will not start actual simulation |
| Actual Response | Launch GUI  Didn’t started Sim |
| Result | Pass |
| Comments | N/A |

**SUNNY DAY:**

|  |  |
| --- | --- |
| Test ID | UCTS\_01\_Run Simulation |
| Purpose | To test the run of the simulation in initialization |
| Preconditions | User is able to open project in NetBeans |
| Inputs | variable of type Simulation |
| Expected Response | JUNIT- PASS:  GUI - Launch  start actual simulation |
| Actual Response | Launch GUI  started Sim |
| Result | Pass |
| Comments | N/A |

# 

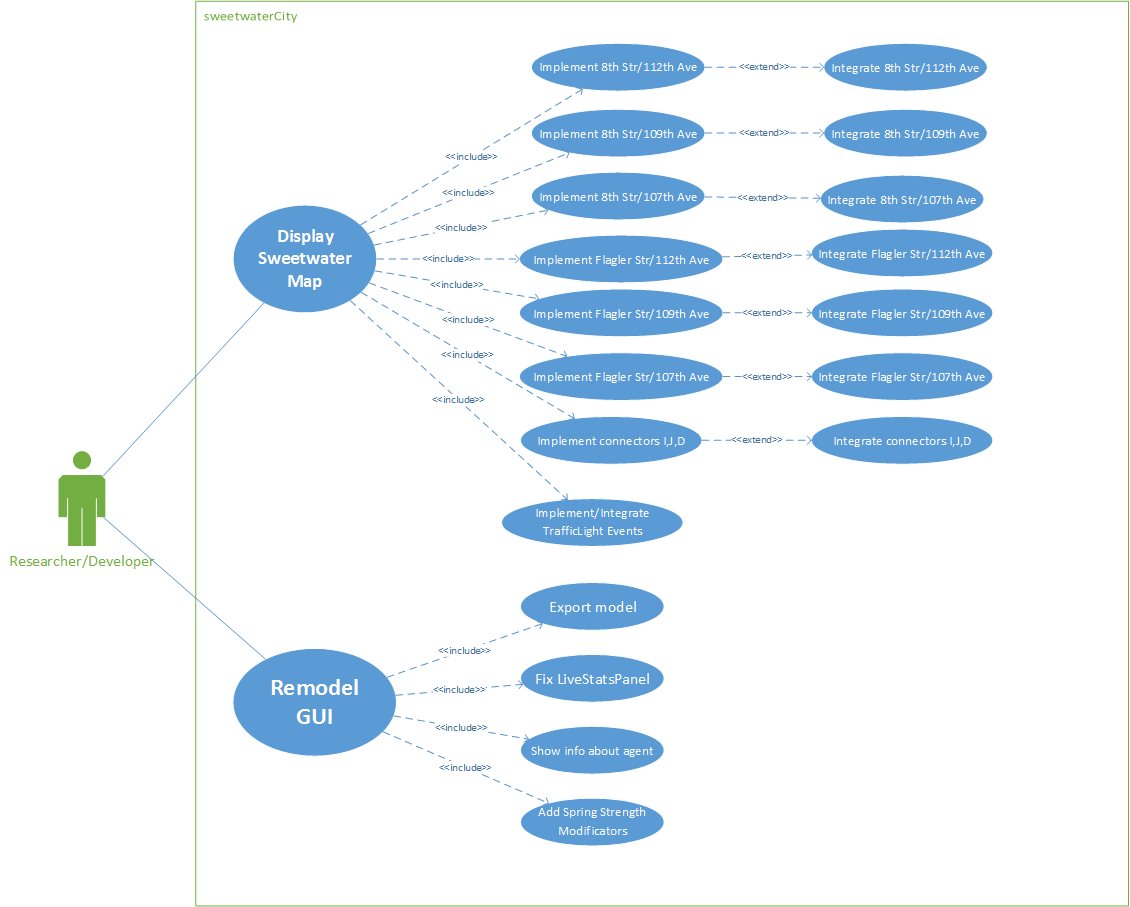
# 

# 

# **Appendix**

## Appendix A - UML Diagrams

## Static UML Diagrams

**Use Case Diagram**

**Use Cases**

**USE CASE ID:** IT0048 Add export function **Level:** High Level

**Details:**

**Actors** - User

**Preconditions**

1. Simulation is running

**Description**

1. The actor begins by clicking on a button in the top middle of the screen. (Trigger)
2. The system responds by loading an editable tab to the user.
3. User provides file name, path, and desired file type.

**Post-Conditions**

1. A visual representation of the model exists.

**Exceptions**

n/a

**Decision Support:**

Frequency – None.

Critically – High.

Risk – Medium

**Constraints:**

* Not too many available file types.

**Usability:**

* No previous training time required

**Reliability:**

* Mean time to failure – 1% failures for every twenty four hours is acceptable.
* Availability – Always.

**Performance:**

The system should have the export performed immediately

**Supportability:**

* System should deliver clear request.

**Modification History:**

Owner: IT Senior Project  
Initiation Date: 7/16/2015

Date last modified: 7/20/2015

**USE CASE ID:** IT0049 See live stats panel **Level:** High Level

**Details:**

**Actors** - User

**Preconditions**

1. Simulation is running

**Description**

1. The actor Looks at the livestats panel.

**Post-Conditions**

1. A visual representation of agent behavior exists on live stats panel.

**Exceptions**

n/a

**Decision Support:**

Frequency – Often.

Critically – High.

Risk – Medium

**Constraints:**

* None.

**Usability:**

* No previous training time required

**Reliability:**

* Mean time to failure – 1% failures for every twenty four hours is acceptable.
* Availability – Always.

**Performance:**

The system should have the export performed immediately

**Supportability:**

* System should deliver clear request.

**Modification History:**

Owner: IT Senior Project  
Initiation Date: 7/20/2015

Date last modified: 7/22/2015

**USE CASE ID:** IT0050 Show info about agent on click **Level:** High Level

**Details:**

**Actors** - User

**Preconditions**

1. Simulation is running

**Description**

1. The actor begins by clicking on a agent. (Trigger)
2. The system responds by loading an panel with agent stats.

**Post-Conditions**

1. Agent stats is visible.

**Exceptions**

n/a

**Decision Support:**

Frequency – low.

Critically – High.

Risk – Medium

**Constraints:**

* Agents are moving and hard to click.

**Usability:**

* No previous training time required

**Reliability:**

* Mean time to failure – 1% failures for every twenty four hours is acceptable.
* Availability – Always.

**Performance:**

The system should have the export performed immediately

**Supportability:**

* System should deliver clear request.

**Modification History:**

Owner: IT Senior Project  
Initiation Date: 7/16/2015

Date last modified: 7/20/2015

**USE CASE ID:** IT0051 Add spring strength **Level:** High Level

**Details:**

**Actors** - User

**Preconditions**

1. Simulation is running

**Description**

1. The actor begins by clicking on a button in the top left of the screen. (Trigger)
2. The system responds by loading an a menu box for each spring.
3. User changes spring strength.

**Post-Conditions**

1. spring strength modified.

**Exceptions**

n/a

**Decision Support:**

Frequency – low.

Critically – High.

Risk – Medium

**Constraints:**

* none.

**Usability:**

* No previous training time required

**Reliability:**

* Mean time to failure – 1% failures for every twenty four hours is acceptable.
* Availability – Always.

**Performance:**

The system should have the export performed immediately

**Supportability:**

* System should deliver clear request.

**Modification History:**

Owner: IT Senior Project  
Initiation Date: 7/16/2015

Date last modified: 7/20/2015

**USE CASE ID:** IT0048 Export function **Level:** High Level

**Details:**

**Actors** - User

**Preconditions**

1. Simulation is running

**Description**

1. The actor begins by clicking on a button in the top middle of the screen. (Trigger)
2. The system responds by loading an editable tab to the user.
3. User provides file name, path, and desired file type.

**Post-Conditions**

1. A visual representation of the model exists.

**Exceptions**

n/a

**Decision Support:**

Frequency – None.

Critically – High.

Risk – Medium

**Constraints:**

* Not too many available file types.

**Usability:**

* No previous training time required

**Reliability:**

* Mean time to failure – 1% failures for every twenty four hours is acceptable.
* Availability – Always.

**Performance:**

The system should have the export performed immediately

**Supportability:**

* System should deliver clear request.

**Modification History:**

Owner: IT Senior Project  
Initiation Date: 7/16/2015

Date last modified: 7/20/2015

**USE CASE ID:** IT0030 Implement 8th st 112th av **Level:** High Level

**Details:**

**Actors** - Developer

**Preconditions**

1Lane graph is complete and accurate

**Description**

1. The actor begins by hard coding necessary objects.

**Post-Conditions**

1. A visual representation of the model exists in the GUI.

**Exceptions**

n/a

**Decision Support:**

Frequency – None.

Critically – High.

Risk – Medium

**Constraints:**

* Accuracy of information, human error.

**Usability:**

* No previous training time required

**Reliability:**

* Mean time to failure – 1% failures for every twenty four hours is acceptable.
* Availability – Always.

**Performance:**

The system should have the export performed immediately

**Supportability:**

* System should deliver clear request.

**Modification History:**

Owner: IT Senior Project  
Initiation Date: 6/16/2015

Date last modified: 6/20/2015

**USE CASE ID:** IT0031 Implement 8th st 109th av **Level:** High Level

**Details:**

**Actors** - Developer

**Preconditions**

Lane graph is complete and accurate

**Description**

1. The actor begins by hard coding necessary objects.

**Post-Conditions**

1. A visual representation of the model exists in the GUI.

**Exceptions**

n/a

**Decision Support:**

Frequency – None.

Critically – High.

Risk – Medium

**Constraints:**

* Accuracy of information, human error.

**Usability:**

* No previous training time required

**Reliability:**

* Mean time to failure – 1% failures for every twenty four hours is acceptable.
* Availability – Always.

**Performance:**

The system should have the export performed immediately

**Supportability:**

* System should deliver clear request.

**Modification History:**

Owner: IT Senior Project  
Initiation Date: 6/17/2015

Date last modified: 6/21/2015

**USE CASE ID:** IT0032 Implement 8th st 107th av **Level:** High Level

**Details:**

**Actors** - Developer

**Preconditions**

1. Lane graph is complete and accurate

**Description**

1. The actor begins by hard coding necessary objects.

**Post-Conditions**

1. A visual representation of the model exists in the GUI.

**Exceptions**

n/a

**Decision Support:**

Frequency – None.

Critically – High.

Risk – Medium

**Constraints:**

* Accuracy of information, human error.

**Usability:**

* No previous training time required

**Reliability:**

* Mean time to failure – 1% failures for every twenty four hours is acceptable.
* Availability – Always.

**Performance:**

The system should have the export performed immediately

**Supportability:**

* System should deliver clear request.

**Modification History:**

Owner: IT Senior Project  
Initiation Date: 6/18/2015

Date last modified: 6/22/2015

**USE CASE ID:** IT0033 Implement Flagler 112th av **Level:** High Level

**Details:**

**Actors** - Developer

**Preconditions**

1Lane graph is complete and accurate

**Description**

1. The actor begins by hard coding necessary objects.

**Post-Conditions**

1. A visual representation of the model exists in the GUI.

**Exceptions**

n/a

**Decision Support:**

Frequency – None.

Critically – High.

Risk – Medium

**Constraints:**

* Accuracy of information, human error.

**Usability:**

* No previous training time required

**Reliability:**

* Mean time to failure – 1% failures for every twenty four hours is acceptable.
* Availability – Always.

**Performance:**

The system should have the export performed immediately

**Supportability:**

* System should deliver clear request.

**Modification History:**

Owner: IT Senior Project  
Initiation Date: 6/16/2015

Date last modified: 6/20/2015

**USE CASE ID:** IT0034 Implement Flagler 109th av **Level:** High Level

**Details:**

**Actors** - Developer

**Preconditions**

Lane graph is complete and accurate

**Description**

1. The actor begins by hard coding necessary objects.

**Post-Conditions**

1. A visual representation of the model exists in the GUI.

**Exceptions**

n/a

**Decision Support:**

Frequency – None.

Critically – High.

Risk – Medium

**Constraints:**

* Accuracy of information, human error.

**Usability:**

* No previous training time required

**Reliability:**

* Mean time to failure – 1% failures for every twenty four hours is acceptable.
* Availability – Always.

**Performance:**

The system should have the export performed immediately

**Supportability:**

* System should deliver clear request.

**Modification History:**

Owner: IT Senior Project  
Initiation Date: 6/17/2015

Date last modified: 6/21/2015

**USE CASE ID:** IT0035 Implement Flagler 107th av **Level:** High Level

**Details:**

**Actors** - Developer

**Preconditions**

1. Lane graph is complete and accurate

**Description**

1. The actor begins by hard coding necessary objects.

**Post-Conditions**

1. A visual representation of the model exists in the GUI.

**Exceptions**

n/a

**Decision Support:**

Frequency – None.

Critically – High.

Risk – Medium

**Constraints:**

* Accuracy of information, human error.

**Usability:**

* No previous training time required

**Reliability:**

* Mean time to failure – 1% failures for every twenty four hours is acceptable.
* Availability – Always.

**Performance:**

The system should have the export performed immediately

**Supportability:**

* System should deliver clear request.

**Modification History:**

Owner: IT Senior Project  
Initiation Date: 6/18/2015

Date last modified: 6/22/2015

**USE CASE ID:** IT0020 Integrate 8th st 112th av **Level:** High Level

**Details:**

**Actors** - Developer

**Preconditions**

Implementation is complete.

**Description**

1. The actor begins by testing the implementation of the objects.
2. Fixes all errors.

**Post-Conditions:**

Agents are able to use the model as it exists in the GUI.

**Exceptions**

n/a

**Decision Support:**

Frequency – once.

Critically – High.

Risk – Medium

**Constraints:**

* Accuracy of information, human error.

**Usability:**

* No previous training time required

**Reliability:**

* Mean time to failure – 1% failures for every twenty four hours is acceptable.
* Availability – Always.

**Performance:**

The system should have the export performed immediately

**Supportability:**

* System should deliver clear request.

**Modification History:**

Owner: IT Senior Project  
Initiation Date: 6/25/2015

Date last modified: 6/30/2015

**USE CASE ID:** IT0021 Integrate 8th st 107th av **Level:** High Level

**Details:**

**Actors** - Developer

**Preconditions**

Implementation is complete.

**Description**

1. The actor begins by testing the implementation of the objects.
2. Fixes all errors.

**Post-Conditions:**

Agents are able to use the model as it exists in the GUI.

**Exceptions**

n/a

**Decision Support:**

Frequency – once.

Critically – High.

Risk – Medium

**Constraints:**

* Accuracy of information, human error.

**Usability:**

* No previous training time required

**Reliability:**

* Mean time to failure – 1% failures for every twenty four hours is acceptable.
* Availability – Always.

**Performance:**

The system should have the export performed immediately

**Supportability:**

* System should deliver clear request.

**Modification History:**

Owner: IT Senior Project  
Initiation Date: 6/25/2015

Date last modified: 6/30/2015

**USE CASE ID:** IT0023 Integrate 8th 109th av **Level:** High Level

**Details:**

**Actors** - Developer

**Preconditions**

Implementation is complete.

**Description**

1. The actor begins by testing the implementation of the objects.
2. Fixes all errors.

**Post-Conditions:**

Agents are able to use the model as it exists in the GUI.

**Exceptions**

n/a

**Decision Support:**

Frequency – once.

Critically – High.

Risk – Medium

**Constraints:**

* Accuracy of information, human error.

**Usability:**

* No previous training time required

**Reliability:**

* Mean time to failure – 1% failures for every twenty four hours is acceptable.
* Availability – Always.

**Performance:**

The system should have the export performed immediately

**Supportability:**

* System should deliver clear request.

**Modification History:**

Owner: IT Senior Project  
Initiation Date: 6/25/2015

Date last modified: 6/30/2015

**USE CASE ID:** IT0020 Integrate Flagler st 112th av **Level:** High Level

**Details:**

**Actors** - Developer

**Preconditions**

Implementation is complete.

**Description**

1. The actor begins by testing the implementation of the objects.
2. Fixes all errors.

**Post-Conditions:**

Agents are able to use the model as it exists in the GUI.

**Exceptions**

n/a

**Decision Support:**

Frequency – once.

Critically – High.

Risk – Medium

**Constraints:**

* Accuracy of information, human error.

**Usability:**

* No previous training time required

**Reliability:**

* Mean time to failure – 1% failures for every twenty four hours is acceptable.
* Availability – Always.

**Performance:**

The system should have the export performed immediately

**Supportability:**

* System should deliver clear request.

**Modification History:**

Owner: IT Senior Project  
Initiation Date: 6/25/2015

Date last modified: 6/30/2015

**USE CASE ID:** IT0021 Integrate Flagler st 107th av **Level:** High Level

**Details:**

**Actors** - Developer

**Preconditions**

Implementation is complete.

**Description**

1. The actor begins by testing the implementation of the objects.
2. Fixes all errors.

**Post-Conditions:**

Agents are able to use the model as it exists in the GUI.

**Exceptions**

n/a

**Decision Support:**

Frequency – once.

Critically – High.

Risk – Medium

**Constraints:**

* Accuracy of information, human error.

**Usability:**

* No previous training time required

**Reliability:**

* Mean time to failure – 1% failures for every twenty four hours is acceptable.
* Availability – Always.

**Performance:**

The system should have the export performed immediately

**Supportability:**

* System should deliver clear request.

**Modification History:**

Owner: IT Senior Project  
Initiation Date: 6/25/2015

Date last modified: 6/30/2015

**USE CASE ID:** IT0023 Integrate Flagler 109th av **Level:** High Level

**Details:**

**Actors** - Developer

**Preconditions**

Implementation is complete.

**Description**

1. The actor begins by testing the implementation of the objects.
2. Fixes all errors.

**Post-Conditions:**

Agents are able to use the model as it exists in the GUI.

**Exceptions**

n/a

**Decision Support:**

Frequency – once.

Critically – High.

Risk – Medium

**Constraints:**

* Accuracy of information, human error.

**Usability:**

* No previous training time required

**Reliability:**

* Mean time to failure – 1% failures for every twenty four hours is acceptable.
* Availability – Always.

**Performance:**

The system should have the export performed immediately

**Supportability:**

* System should deliver clear request.

**Modification History:**

Owner: IT Senior Project  
Initiation Date: 6/25/2015

Date last modified: 6/30/2015

**Class Diagrams**

## 

## 

## Figure 1: Entity component(package) - Minimal Class Diagram

## Figure 2: Events component(package) - Class Diagram

## 

**Figure 3: GUI component(package) - Class Diagram**

## 

## 

## Figure 4: Organization component(package) - Class Diagram

## 

## 

**Figure 5: Simulation component(package) - Class Diagram**

## 

## Figure 5: Simulation component(package) - Class Diagram

## 

## Figure 6: SweetwaterCity component(package) - Class Diagram

## 

**Figure 7: Tests component(package) - Class Diagram**

## Figure 8: Barbarossaplatz component(package) - Class Diagram

## 

**Figure D - ER diagram for Entity package**

…

### 

### 

### 

### Dynamic UML Diagrams

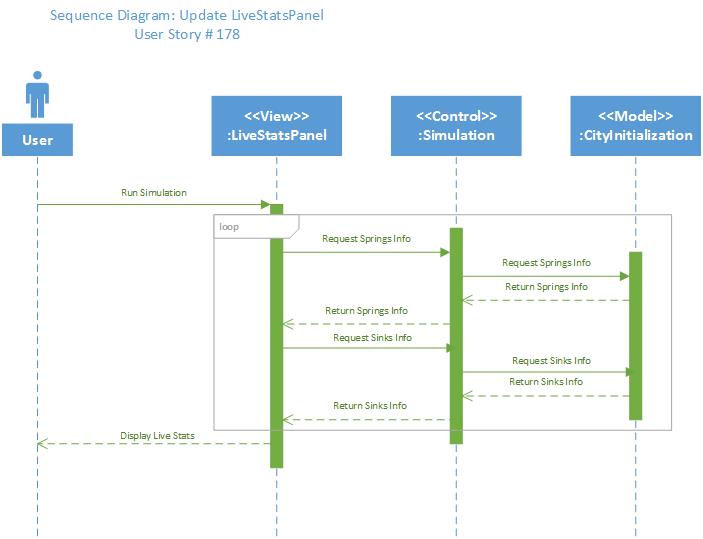
### 

### 

**Figure 9: Run Simulation - Sequence Diagram**

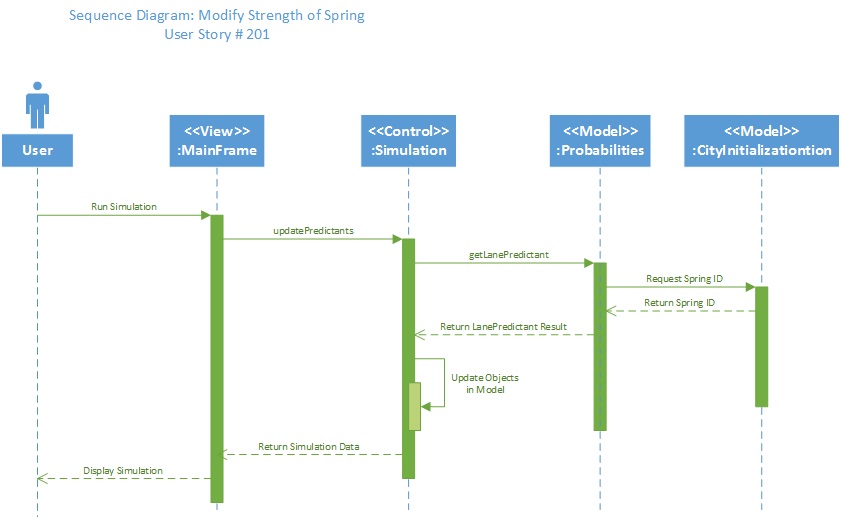
### 

**Figure 10: Export Function (User Story #190) - Sequence Diagram**



**Figure 11: Update LiveStatsPanel (User Story #178) - Sequence Diagram**

## 

**Figure 12: Modify Strength of Spring (User Story #201) - Sequence Diagram**

## 

## 

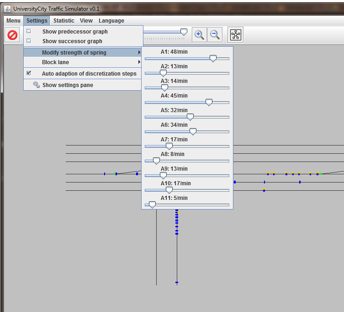
## 

## 

## Appendix B - User Interface Design

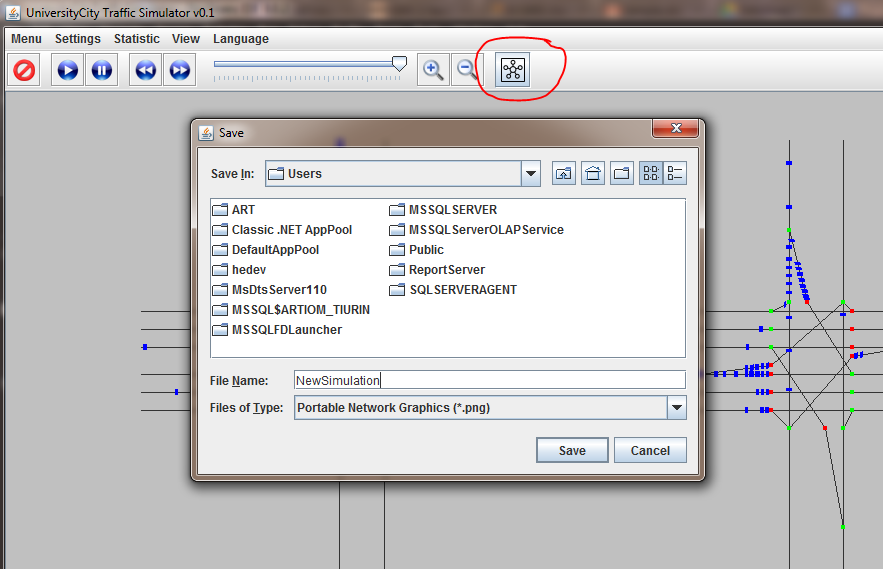
**Spring Strength Modificators**:

Additional modificators were added to modify all Springs



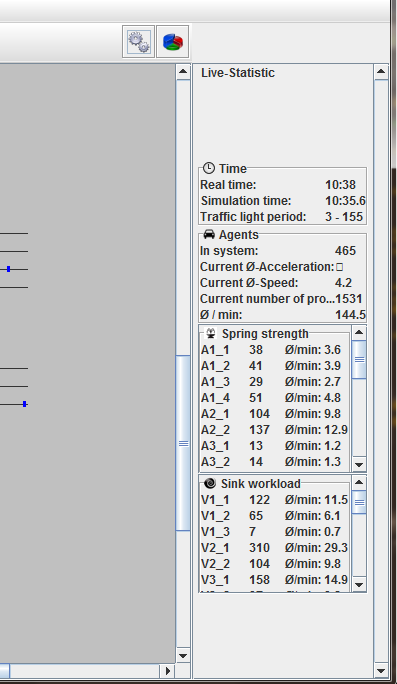
**Export Screen Shot of The Model:**

Now with a click of the button a screenshot of the can be made and exported as PNG or JPEG file.



**Spring Strength and Sink workload:**

Live-Statistics panel updated for user to see all Springs and Sinks.



## 

## 

## Appendix C - Sprint Review Reports

**Sprint 1 Report**

**Date:** 05/29/15

**Attendees:** Artiom Tiurin, Daniel Costa e sa, Charaf Azzouzi, Gerald Inberg.

**Discussed Topics:**

-Researched tools for building diagrams for base code

-Created Class diagrams for Barbarossaplatz (Base Code)

-Get more familiarized with Base Code

-Realization that a lot of User Stories were actually Epics. Therefore, moved to Backed Log and refined.

**Sprint 2 Report**

**Date:** 06/12/15

**Attendees:** Artiom Tiurin, Daniel Costa e sa, Charaf Azzouzi, Gerald Inberg.

**Discussed Topics:**

-Starting points in advancing to create Sweetwater model

-initial steps has been taken. Data started to get collected from Sweetwater map

-Sketch Graph, Traffic Lane Graph have been drawn in Visio.

-Lane measurements were taken from Visio to GUI.

-First tries to implement a small example related to Sweetwater city.

**Sprint 3 Report**

**Date:** 06/26/15

**Attendees:** Artiom Tiurin, Daniel Costa e sa, Charaf Azzouzi, Gerald Inberg.

**Discussed Topics:**

-After first attempt was unsuccessful to present working partial model, errors were found.

-Data: Lane coordinates were mismatched with GUI panel coordinate. Redone

-Initial Implementation of the part of the model was fixed.

-Factor of human error played a lot in accomplishing tedious tasks.

-Many objects required for model were implemented. integration was lagging ther demo.

**Sprint 4 Report**

**Date: 7/10/15**

**Attendees:** Artiom Tiurin, Daniel Costa e sa

**Discussed Topics:**

-Work has been done for the past Sprint. Most user stories were started but not completed to the end.

-A lot of planned work fell behind.

-Demo has not changed

-Long weekend(4th of July) put a restriction on collecting real data, thus delaying the implementation

-Intersections/junctions/Dissections were implemented according to the system’s functionality

## Sprint 5 Report

## Date: 7/24/15

## Attendees: Artiom Tiurin, Daniel Costa e sa

## Discussed Topics:

-Demo of completed User Stories: Traffic Light Events,

-Logic of the for the city of Sweetwater modified by directions in A-,M-,K-sections

-Additions to GUI: Spring Strength Modificators, aligned Springs and Sinks in Statistic panel.

-Overview of the project contribution for the whole semester.

-Overview of work done from Product Owner’s perspectives.

## 

## 

## 

## Appendix D - Sprint Retrospective Reports

**Sprint 1 Retrospective**

**Date:** 5/29/2015

**Attendees:** Artiom Tiurin , Daniel Costa e sa

**Discussed Topics:**

-After accepting a Base Code from Product Owners, it took time to get around in understanding the system.

-Spend more time trying to understand the code.

-Communicate with Product Owners for any obstacles.

**Sprint 2 Retrospective**

**Date:** 6/12/2015

**Attendees:** Artiom Tiurin , Daniel Costa e sa

**Discussed Topics:**

-Need more time to invest into accomplishing a presentable Demo.

-Allow more time for tedious work.

-Do not fall behind on assigned user stories.

-Understand the velocity of a user story(projected time) and tasks(actual time spent) of the project.

-Communicate more with Product Owners for any obstacles.

**Sprint 3 Retrospective**

**Date:** 6/26/2015

**Attendees:** Artiom Tiurin , Daniel Costa e sa

**Discussed Topics:**

-Sprint 3 was a good sprint in terms of accomplishing presentable demo.

-eep the same work speed.

-Keep up with documentation.

-Listen to what Prof. Masoud is advising.

**Sprint 4 Retrospective**

**Date:** 7/10/2015

**Attendees:** Artiom Tiurin, Daniel Costa e sa

**Discussed Topics:**

First student: Artiom Tiurin

-Fell a little short, per Product Owners

-Make sure to work on one user story at the time.

-Improve Demo for the end of next Sprint 5 and final presentation.

-Work on Documentation and update Prof. Masoud and Product Owners on progress

* Use Cases
* Seq. Diagrams
* Class Diagrams

-Mark review notes for other students

-First of Sprint 5: finish user stories of Sprint 4

Second student: Daniel Costa e sa

-Fell short this sprint, per Product Owners

-Update documentation and update Product Owners and Prof. Masoud on progress

-Have a working Demo for each Sprint Review presentation and especially for the last Sprint 5.

-Keep up with Daily Scrums, Mingle and communicate obstacles to Product Owners.

-Keep calm during the presentations/interviews, no matter what happens.

-First of Sprint 5: finish user stories of Sprint 4.

**Sprint 5 Retrospective**

**Date:** 7/24/2015

**Attendees:** Artiom Tiurin, Daniel Costa e sa

**Discussed Topics:**

First student: Artiom Tiurin

-Documentation should have been done along the way with Sprints. As a general rule, documentation should help development process. The end of semester will not be overwhelmed with documentation.

Second student: Daniel Costa e sa

-Must follow the development process: UserStory/UseCase, Class Diagram, Sequence Diagram, Implementation, Validation.

-As a general rule, documentation should help development process, i.e. Seq. Diagrams prior Implementation.

# 

# 

# 

# 

# **References**

Traffic Light Data, Miami-Dade County: <http://www.md-atms.net/TrafficOpsMenu.aspx>