**iTraffic**

**A Traffic Simulator that will deliver dynamic traffic flow through dynamic traffic lights and proper time allocation**

A Project Study Presented to the Faculty of the College of Information,

Computer and Communications Technology

University of San Jose – Recoletos

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

In Partial Fulfillment of the

Prelim Requirements for Thesis 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Mel Raymond G. Salma

*Proponent*

Mr. John Leeroy Gadiane

*Adviser*

**CHAPTER I**

**INTRODUCTION**

**Rationale of the Study**

Nowadays, the streets of Cebu are witnessing an unprecedented increase in traffic congestion due to road repairs, road accidents, and most especially inadequate traffic light systems. The two factors namely: road repairs and road accidents are events that can’t be controlled or handled completely by traffic personnel, but unlike those two the traffic light systems are. The problem with the current traffic light systems of today is that the traffic agent handling the traffic lights of a certain traffic location can’t discern properly the sequence of the timer and light allocation for the congested traffic present. There are certain ways that these agents handle a congested traffic situation. One of which would be on setting the traffic timer either manually or pre-set. And the other is by making use of the traffic road agents who are present at the congested traffic location, in order to formulate a judgment on what to do with the congested traffic situation.

In one of the major cities of Cebu, namely in Mandaue City, it is known for its advanced traffic system along with a large number of traffic cameras at various road intersection points and a data centre to boot. Mandaue’s main force for monitoring the entire traffic and road situation in the city is called the Traffic Enforcement Agency of Mandaue (TEAM). Recently, the TEAM has currently been using an application that delivers manual or pre-set traffic timer system that will handle every road intersection found in the city, according to the TEAM head. Currently, the application has performed suitably with regards to handling the traffic flow with its traffic light and timer allocation, but it doesn’t aid the traffic agent with analysis concerning the congested traffic event. So by analysing the entirety of the situation to how it became congested, along with the amount of time that was given to lessen or eliminate the congested traffic it would prove to be useful for the traffic agent for future events. With other regards to the app, it works in such that the traffic agent is capable of pre-setting the traffic timers on a given road intersection and would also be able to manually configure them to whichever would benefit the road. Now at the occurrence of congested traffic and it seems to be going nowhere, a traffic road agent would take into action to contact a traffic agent at the data centre to configure the existing traffic timers and lights in order to lessen the congested traffic. Here, the traffic agent would then be making use of his training on how much time to give, in order to lessen the congestion.

The current traffic system in Cebu, or anywhere in the Philippines, is mainly composed of a traffic enforcer, traffic lights with a traffic counter, and a data centre. The traffic enforcer is usually seen on the road if either the traffic light/s are malfunctioning or the congested traffic is too much or going too long. The traffic enforcer would be considered the front-face of the traffic management system of the city, and at the back-end of it, all would be the data centre with its respective agents who are pulling the strings. Here, they will be manipulating the traffic light system, in order to fix or create a solution to the congested traffic that is occurring or is simply pre-setting the traffic lights for a balanced traffic flow. But the traffic system, so far, comes with limitations with regard to different traffic situations that would be progressing in the future. These limitations are that:

1. The traffic system only considers the current situation that is occurring on the spot. It only focuses on what’s in front of the traffic and not take into consideration those who are following.
2. The traffic enforcers can’t balance the traffic altogether or remove the biases that they have discerned through normal vision. So what is totally ongoing at the current traffic situation because of the fact that the enforcer has so little vision and he can’t weigh in concerning the biases and decision-making that need to take place, and
3. The traffic counter isn’t automated as it only possesses a set amount of time, or even if the current traffic isn’t that heavy, the counter remains the same because it is being set at a predefined amount for that particular lane or traffic.

The goal of this research would be to develop an application that will serve as an extra insight for the traffic agent on what to do for a certain congested traffic situation. That insight would be beneficial, so as to remove any biases that the traffic agent might be afflicted upon. Now, to what application that could assist the traffic agent in such a way would be a **traffic simulator.** By using a traffic simulator the user would then be able to duplicate a congested traffic event and is free to add multiple inputs like vehicles of different speed and type and also road regulations. And during or after a traffic run simulation, the agent would then be presented with analyzed data that would be collected by the simulator. The simulator would be capable of presenting such data by analyzing both the speed and quantity of the vehicles during the simulation. So with the data collected, the traffic agent would now be able to provide an unbiased and processed judgment to delivering an adequate or proper allocation or transition of traffic light along with its respective timer.

**THEORETICAL BACKGROUND**

**Using a Simulator for Traffic Duplication**

There has been an ever-growing issue with regard to properly managing congested traffic here in Cebu. Mainly, what has been causing such congested traffic would be noticeably, the inadequate transition of the traffic timer and lights.

Most of these issues that lead to traffic congestion are unmanageable there are however those that are.

Traffic agents that handle the traffic light system are the ones feeling the most pressure with handling the

With the problem revolving around the current traffic system which is mainly its traffic light system. The researcher will be undergoing a study that aims to create a representation of a traffic situation and use that data to create a dynamic traffic system to create a smooth transition of traffic. By doing so will aid the agent by removing biases, and this would mean that the solution to that problem must be capable of presenting doable or believable outputs which will enlighten the user. And such data handling and data representation can be made done in with the use of Computer Science.

Computer Science is a study that is all about problem-solving, and it possesses branches that

Computer Science is a field of with, and as such, any problem can be solvable with the right parameters and the right algorithm to go along with it. Computer Science is all about problems and problem solving, and it is up to the researcher to create an answer to the problem [1]. Now, the traffic agent is need of a tool that would be able to assist them in providing insights on what to do for a certain congested traffic situation. To create that insight it would require a tool capable of reduplicating that situation so that the agent can see with his own eyes the actual flow of what’s going on.

That tool is what many engineers would call a ***simulator.*** A simulator is a very useful tool that is used to create an imitation or a reconstruction of any certain events in order to create an assessment or analysis [2]. With the use of a traffic simulator, the traffic agent will be able to make use of models from actual roads in traffic, so by recreating a congested traffic situation would be made very visible with the aid of it.

**Traffic Light System**

**MACHINE LEARNING**

**REVIEW OF RELATED STUDIES**

**Improving Traffic Light System**

**Smart Traffic Light Control System**

**Combining Simulation with Machine Learning and Data Mining**

**SUMO – Simulation of Urban Mobility**

**PROJECT OBJECTIVES**

The study aims to develop a traffic simulator which will aid in the process of reducing the traffic congestion on the streets of Mandaue City.

More specifically it aims to:

1. Simulate a traffic scenario
2. Make use of the Mandaue Road and Traffic System
3. Automate traffic lights and time allocation
4. Develop a dynamic time allocator of traffic lights based on traffic volume

**PROJECT SCOPE AND LIMITATION**

**RESEARCH METHODOLOGY**

**CHAPTER II**

**SOFTWARE REQUIREMENTS AND DESIGN SPECIFICATIONS**

This chapter describes the requirements that the study intends to accomplish. It also presents the processes of the system to be able to achieve the said requirements. This chapter includes the Use Case Model, Use Case Narratives, Activity Diagrams, and Class Diagram to illustrate the design of the system.

**USE CASE DIAGRAM**

User

System

*Figure 4. Use Case Model*

The diagram above represents the use case diagram of the study. The user of the simulator will select a location for the simulation and after the user will input vehicles with their corresponding properties. These properties being one of them is the speed of the vehicles. After the user will start the simulation, the output will be the inputted car/s and traffic light/s that will re-enact an actual traffic situation.

**Use Case Narrative: UC 01 Location Selection and Build Map**

|  |  |
| --- | --- |
| Use Case | Select location for simulation |
| Actors: | Traffic personnel |
| Purpose: | Place for the simulation |
| Overview: | This method will allow the user to select the location for the whole simulation to take place, and would then be able to build the map for simulation |
| Type: | Essential |
| Precondition: | Access granted by the traffic officers |
| Postcondition: | A smooth traffic flow transition will be displayed |
| Flow of Events | |
| Actor Action | System Response |
| Input location for simulation  Build map | 1. The user will be asked for a location input for anywhere in Mandaue City at a input field at the sidebar navigation menu. |
| 1. It will display the area itself via OSM |
| 1. After the map interface is displayed, the action to add input/s of cars and/or traffic light/s will now be made accessible at the sidebar navigation menu. |

**Use Case Narrative: UC 02 Vehicle Attribute Input**

|  |  |
| --- | --- |
| Use Case | Input vehicle attributes |
| Actors: | Traffic personnel |
| Purpose: | For analyzation and traffic flow purposes |
| Overview: | This is where the user will input a vehicle/s attributes. The newly inputted vehicle will then be available and accessible to be drag n’ dropped to the maps. |
| Type: | Essential |
| Precondition: | User needs to input a location for simulation |
| Postcondition: | User will be able to add vehicles to the simulator |
| Flow of Events | |
| Actor Action | System Response |
| Add vehicle | 1. The system will ask for input regarding the vehicle |
| 1. After, the vehicle inputted will be displayed for use. |
|  |

**Use Case Narrative: UC 03 Starting Traffic Sequence**

|  |  |
| --- | --- |
| Use Case | Start Traffic Sequence |
| Actors: | Traffic personnel |
| Purpose: | To produce dynamic time allocation of traffic lights |
| Overview: | This will produce the output of the simulator, the dynamic traffic counter and traffic lights |
| Type: | Essential |
| Precondition: | Vehicles must be inputted to lane/s |
| Postcondition: | Output will be released |
| Flow of Events | |
| Actor Action | System Response |
| Starting the traffic sequence | 1. The user will be displayed with the traffic sequence |
| 1. System will display the output after the traffic sequence is finished |
|  |

**Use Case Narrative: UC 04 Inputting Street Rule/s to Vehicle**

|  |  |
| --- | --- |
| Use Case | Adding Street Rules to Simulator |
| Actors: | Traffic personnel |
| Purpose: | For training the simulator with actual rules from the streets |
| Overview: | This will add that “real-life” situation feel to the simulator which is by adding actual street rules, and by adjusting the simulator with the inputted rules given like “one-way”, “no left turn”. |
| Type: | Essential |
| Precondition: | Vehicles may or may not yet be inputted to lane/s |
| Postcondition: | Simulation will run with these rules in play |
| Flow of Events | |
| Actor Action | System Response |
| Adding a Street rule for a particular lane | 1. The vehicles will be notified about prescribed rule through event handlers or a flag handler through which will be triggered by new input/s from the user. |
| 1. System will take the sign/s into account for the output basis for display. |
|  |

**Use Case Narrative: UC 05 Stop/Pause Traffic Simulation**

|  |  |
| --- | --- |
| Use Case | Stop/Pausing the Traffic Simulation |
| Actors: | Traffic personnel |
| Purpose: | For adding changes to the current simulation or to terminate the entire simulation to start over |
| Overview: | This will give the user the opportunity to start over again or to add either more cars, traffic rules or traffic lights to the actual simulation |
| Type: | Optional |
| Precondition: | The simulation must have already ran |
| Postcondition: | The simulator will either start over again or will accept new inputs from the user |
| Flow of Events | |
| Actor Action | System Response |
| Pause Simulation/Stop Simulation | 1. If pause, then the system will accept additional inputs by the user |
| 1. If stop, then the system will show a pop-up window for confirming termination process and start a new simulation |

**ACTIVITY DIAGRAMS**

|  |  |  |  |
| --- | --- | --- | --- |
| Search location for Traffic Simulation | | | |
|  | **User** | **System** | **Output** |
| Build map  Inputs location | Extract parsed OSM nodes  Search in Database  Verify Inputted Location | The OSM Map interface will be built and ready to start the simulation  The name of location will be displayed at the input box |

*Figure 5. Search location and Build map for simulation*

Figure 5 illustrates how the user actor of the system will search for a location to start doing the traffic simulation. The initial requirement would be if that location is saved to the database since roads that are to be made use of need to belong to the Mandaue Traffic Road System, only those that belong may be visited for simulation. Then after inputting a valid location for simulation, the system will redirect from the default location which is Mandaue City Hall and to the traffic intersection.

The locations will be verified if whether they are found in the database for locations to be used in the simulation. The system will already be expecting specific locations for the simulation action, and if the inputted location/s are not saved in the database than a pop-up will be shown with the message informing the user that the inputted location is not saved in the database and it can’t be accessed.

After that, the user is then tasked to press on the “Build map” button to extract the nodes and then so build the map interface.

|  |  |  |  |
| --- | --- | --- | --- |
| Input Vehicles for Traffic Simulation | | | |
|  | **User** | **System** | **Output** |
| Input Vehicles Speed  Input Vehicles Type  Input Vehicles ID | Save to Database | Appear in sidebar menu as an available option |

*Figure 6. Input vehicles for simulation*

Figure 6 illustrates the occurrence when traffic enforcer actor will add an input of vehicle/s to the simulator system. As, the actor will add an input it will automatically be saved to database and then made accessible at the sidebar navigation menu for the drag n’ drop to start simulating.

|  |  |  |  |
| --- | --- | --- | --- |
| Drag N’ Drop Vehicles and Input Traffic Lights for Simulation | | | |
|  | **User** | **System** | **Output** |
| Drag n’ drop traffic lights  Drag n’ drop vehicles | Save to Database  Accept dragged input | Traffic Simulation button will be available to click |

*Figure 7. Drag n’ drop vehicles and traffic lights to Traffic Simulator*

Figure 7 illustrates how the system will accept the input into the traffic simulator. Once, a vehicle or traffic light is dragged into a lane it can be deleted/transferred as to what the user prefers. The user is able to do such before starting the simulation, but during the simulation the user can’t affect anymore the placement of the vehicles and traffic lights.

|  |  |  |  |
| --- | --- | --- | --- |
| Simulator System will Process the Traffic Flow | | | |
|  | **User** | **System** | **Output** |
| Start the Simulation | Compute average vehicle speed  Apply Traffic Signal Rules  Determine the best traffic time  Determine LOS  Compute ICU | Smooth Traffic Transition  through Dynamic Time  Allocation |

*Figure 8. Traffic Processing Module*

Figure 8 shows the overall process of the traffic simulator during the actual simulation. The process begins as soon as the traffic enforcer actor will press “Start Simulation”, then the vehicles will move and upon entry to the intersection the ICU LOS will be computed with the factors involving vehicle quantity, vehicle speed and arrival time to intersection. The determining of best traffic time will be based upon lane LOS and with the given basis of amount of time. This will then determine the lane to initially give the right of way. After, the simulator will abide by the traffic signal manual to the amount of time given to transition from current traffic light signal to the next and following to the next lane to be given right of way.

|  |  |  |  |
| --- | --- | --- | --- |
| Traffic Simulator will Display Traffic Report | | | |
|  |  | **System** | **Output** |
|  | Average vehicle speed  Traffic time given  Determined LOS  Computed ICU | Traffic Statistics Report |

*Figure 9. Traffic Simulation Statistics Output*

Figure 9 shows the data that will be taken from the simulator system to be shown to the traffic enforcer actor for analysis. The output will be used to improve traffic flow and traffic management.

|  |  |  |  |
| --- | --- | --- | --- |
| Enabling/Disabling Traffic Light/s to Simulator | | | |
|  | **User** | **System** | **Output** |
| Enable/Disable Traffic Light/s | OSM node Traffic Lights = on/off | Traffic Light/s are enabled/disabled |

*Figure 10. Enable/Disable Traffic Light/s*

Figure 10 shows the process of enabling or disabling the traffic light/s of the simulator. These traffic light/s are already linked to the Open StreetMap API and as such its location found in the simulator will be the same as the exact placement of it in the actual road. The user is able to enable/disable the traffic light/s by simply just pressing on them.

|  |  |  |  |
| --- | --- | --- | --- |
| To Stop/Pause the Traffic Simulation | | | |
|  | **User** | **System** | **Output** |
| Stop the Simulation  Pause the Simulation | Display confirmation alert  Halt motor vehicle/s | An alert box will be shown asking for confirmation to stop the current simulation and to start a new simulation  The current traffic will display motor vehicle/s stop moving |

*Figure 11. Stop/Pause Simulation*

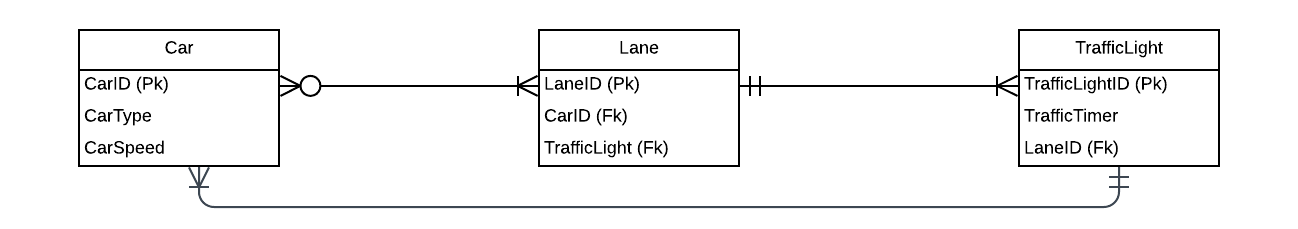
Figure 11 shows the process of when the user will attempt to stop/pause the simulation. At first, if the user wishes to pause the simulation, then the simulator will halt all vehicle/s and the user is allowed to input or add more vehicle/s to the simulator, and is also able to enable/disable traffic light/s. Secondly, if the user wishes to stop the simulation, then an alert window will pop-up asking for a confirmation to stop the simulation and will immediately process to start a new simulation from scratch.

**ENTITY RELATIONSHIP DIAGRAM**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Location | Bounds | | | |
|  | Minlat | Minlong | Maxlat | Maxlong |
| Pacific Mall | 10.3397800 | 123.9450600 | 10.3422100 | 123.9492700 |
| Maguikay Flyover | 10.3338600 | 123.9379900 | 10.3361600 | 123.9422000 |
| U.N. COR. D. M. Cortes St. | 10.3377700 | 123.9531300 | 10.3400700 | 123.9573500 |
| Cor. S.B. Cabahug/A.C. Cortes | 10.3280400 | 123.9438300 | 10.3326400 | 123.9522500 |
| Parkmall Intersection | 10.3237300 | 123.9336900 | 10.3260300 | 123.9379000 |
| S&R Intersection | 10.3179700 | 123.9294300 | 10.3202700 | 123.9336500 |
| Cor. A.S. Fortune/M.C. Briones | 10.3389200 | 123.9440500 | 10.3435200 | 123.9524700 |
| Cor. A.S. Fortune/H. Cortes | 10.3292500 | 123.9463100 | 10.3315500 | 123.9505200 |

*Table 2. List of Intersections found in Mandaue City*

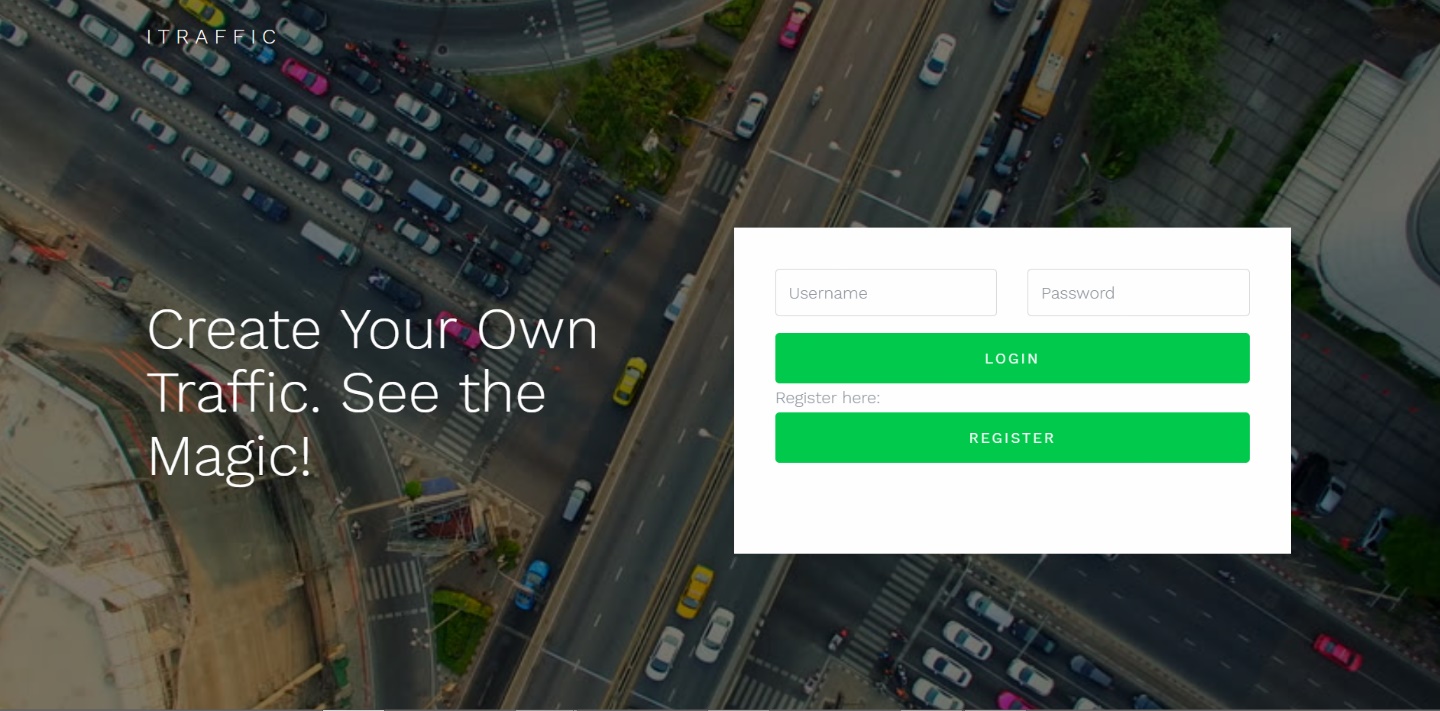
Table 2 presents the table for the intersection locations found in Mandaue City and are to be used for the traffic simulator. These locations will be stored for the simulator to use, and inputted locations that are not from the table will not be accessed.



*Figure 12. Entity Relationship Diagram for the database*

Figure 12 presents the graphical representation of the database structure of the system before simulation. It states that the lane may have an input or zero or many car/s while a car may be directed to one or more lane/s. The traffic light may direct to one or more lane/s while a lane can only have one traffic light to be basing of to. Lastly, a car will follow the sequence of one traffic light and a traffic light may be used to control the sequence of one or more car/s.

**USER INTERFACE DESIGN**

****

*Figure 13. Loading GUI*

Figure 13 illustrates how the app will initially appear as it is being run at first. Here it will show a simple description of what the system is all about.

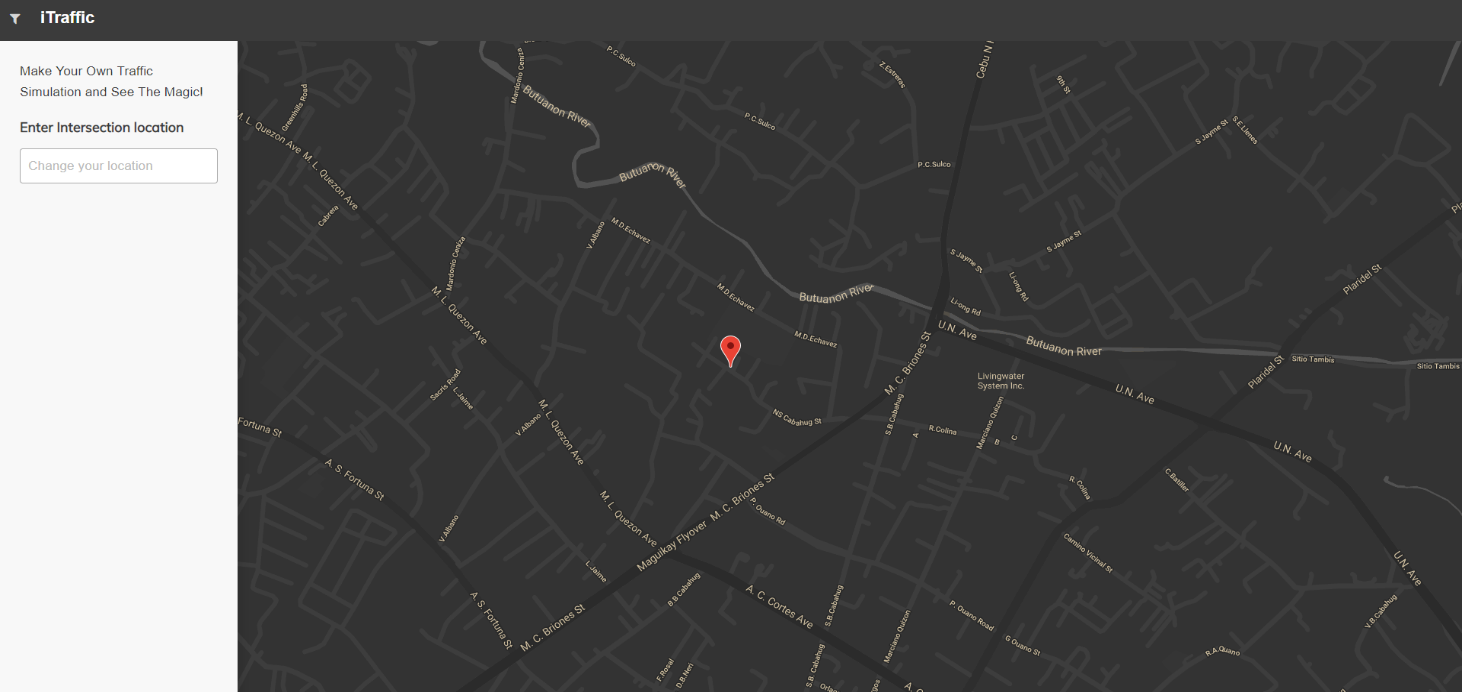
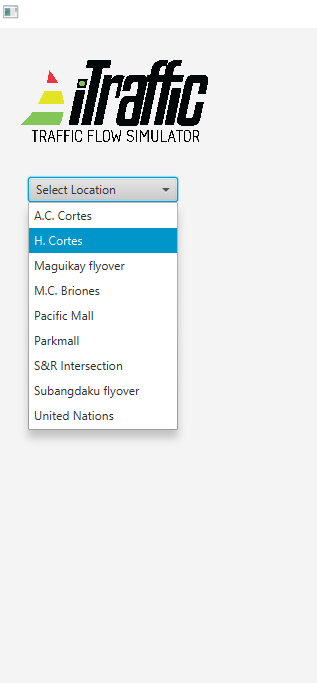
*Figure 14. Initial Simulator Dashboard GUI*

Figure 14 illustrates the landing of the simulator after its preloading phase. Here the maps will show the initial marker which is placed on top of Mandaue’s City Hall.



*Figure 15. Select Location*

Figure 15 illustrates the user interface of selecting a location for simulation. The locations displayed will be of ones which are saved to the database. If a location is inputted and it is not found at the database then the “map” will redirect to the default location and will display a message that the location inputted was not found at the database.

*Figure 16. Input Vehicles*

Figure 16 illustrates how to add attributes for a vehicle to be used by the simulator. In the simulator, the vehicle can be either added or deleted, so the user has full control of what to do and easy to use. The user is expected to input a vehicle’s type and speed for example: “SUV”, “5”, “Automobile”, the speed of the vehicles will be interpreted by k/h (kilometre per hour), if either of the expected attributes are missing than an error will be displayed since the system requires both attributes to be added to comprise a vehicle to be placed at the map simulator. The simulator would make use of the type and speed of the vehicle, in order to analyse them for the traffic light and time allocator algorithm.

*Figure 17. Drag n’ Drop Vehicles to Simulator*

Figure 17 illustrates how to perform drag n’ drop at the system. Drag n’ drop is easy to perform at the system, as such you simply need to click at the icon at the sidebar navigation menu and drop the icon at the lane found at the map interface.

*Figure 18. Drag n’ Drop Traffic Lights to Simulator*

Figure 18 illustrates how to add traffic light/s to the map interface. It basically functions the same as drag n’ dropping a vehicle to the map. The user would just have to click at the traffic light icon and drop it at the location where it is advised to be placed which is at the end of a certain lane because of the condition that vehicle/s are supposed to stop for a while as they await the system to display the traffic light and time allocator.

*Figure 19. Run Simulation*

Figure 19 illustrates the simulation running. The simulator will run as the user will press on the “Run” button at the top right of the frame.

*Figure 20. Stop/Pause Simulation*

Figure 20 illustrates how to stop or pause a simulation. The button for this function is the same one with running the simulator. It is indicated via colour green is “Run Simulation” and red is “Stop/Pause Simulation”.

*Figure 21. Time Allocator through Random Forest Classifier*

Figure 21 illustrates the expected output of the simulator which is the traffic light and time allocator. The simulator will display these via the traffic light and timer. The algorithm behind these outputs are made with the use of the Random Forest Classifier algorithm.

*Figure 22. Add More Vehicles to Simulation*

Figure 22 illustrates how to add more vehicle to the simulator as it is being paused. So, it is easy to perform because it is performed as how it is initially done which is to just press on the “Pause” button found on the top-right corner and then drag the icon of the vehicle you wish to add to the simulator. After you added a vehicle to the simulator, then it will be accepted to the simulator and saved to the database and after the simulator will re-adjust to the newly added data.

*Figure 23. Enabling/Disabling Traffic Light/s*

Figure 23 illustrates how to enable/disable the traffic light/s found at the simulator. It is easily made done by just simply clicking on the traffic light/s that you wish to be turned off/on.

*Figure 24. Presentation of Collected Data from Simulator*

Figure 24 illustrates the presentation of the data collected from the entire simulation which is the average vehicle speed, control delay, LOS, vehicles direction, throughput, time allocated per lane and traffic light respectfully, and lastly the status report of what is generally good or wrong of that certain traffic situation.

**CHAPTER III**

**SOFTWARE DEVELOPMENT AND TESTING**

**Development Software Platforms, Development Environment, and Tools**

The traffic simulation system has two components into it mainly: the simulation front-end and the traffic algorithm back-end. The simulation front-end is where the user will interact with the system to create a traffic simulation. The simulation front-end is developed to be easy-to-use for the user, and this was developed with the use of JavaFX and its Scene Builder. The back-end component is where the system will handle the data to produce an output of an automated traffic light and time allocator. Simply, this is where the data will be collected like the average speed of the vehicles, the LOS of the lane and directions of the vehicle. The system back-end will be processing all of this data and will perform the algorithm that was formulated by the researcher, that the right-of-way or the transition of traffic will be automated whereas, human interaction will still be needed and while the system’s back-end will be handling all of the actual processing, output, analysis and some pre-processing, the user will partake at the initial pre-processing which is at the beginning of running the simulation. For the back-end component, Java is used as the core programming language. The researcher made use of Java mainly because of how platform independent it is and how it can run on various operating systems. The development tool for the Java core is Eclipse IDE 2018-2019 with Java SE using JavaFX and Scene Builder. The database is created using SQLite as the Relational database management system. SQLite is handy and easy to coordinate with JavaSE and it is a simple and clean DMS.

The simulator system’s back-end component, makes use of several external libraries for its development. The simulator is built with the use of OpenStreetMap API (API v0.6, 2009) and JavaFX libraries. The OpenStreetMap API enables the simulator to access actual visualization and location of the locations located in Mandaue City. Since the simulator is intended to automate the traffic road system of Mandaue City, it would need OpenStreetMaps to provide an exact and actual feel of the area since OpenStreetMap is based from live and actual data that was provided from the people from those areas. Also, JavaFX is on the verge of replacing Swing as Java’s default GUI library, and as such to cope with the trend, the researcher is making use of JavaFX because of how more and seemingly advanced it is becoming compared to Swing. JavaFX is used by the traffic simulator system as to be the GUI for the user and as the wrapper for the OpenStreetMap API. Alongside with JavaFX, it has its own separate frame or software for developing the GUI, which is Scene Builder. This software is made or designed as such to sync and work together with JavaFX as its personal yet separate JFrame. Scene Builder is a visual layout tool which is more advanced that Swing’s JFrame. This is a very handle tool which alike JFrame will allows users to drag and drop UI components to a work area. Scene Builder can also allow users to apply style sheets to their GUI so that they can bring out the beauty of their design.

**DEVELOPMENT AND TESTING PROCESS**

**DEVELOPMENT PROCESS**

**Data Retrieval**

The data to be retrieved that will be made used for the simulator would be the location name of the street that would be found at Mandaue City, the vehicle’s attributes mainly width, height, type and speed, vehicle’s lane and direction, and road regulations.

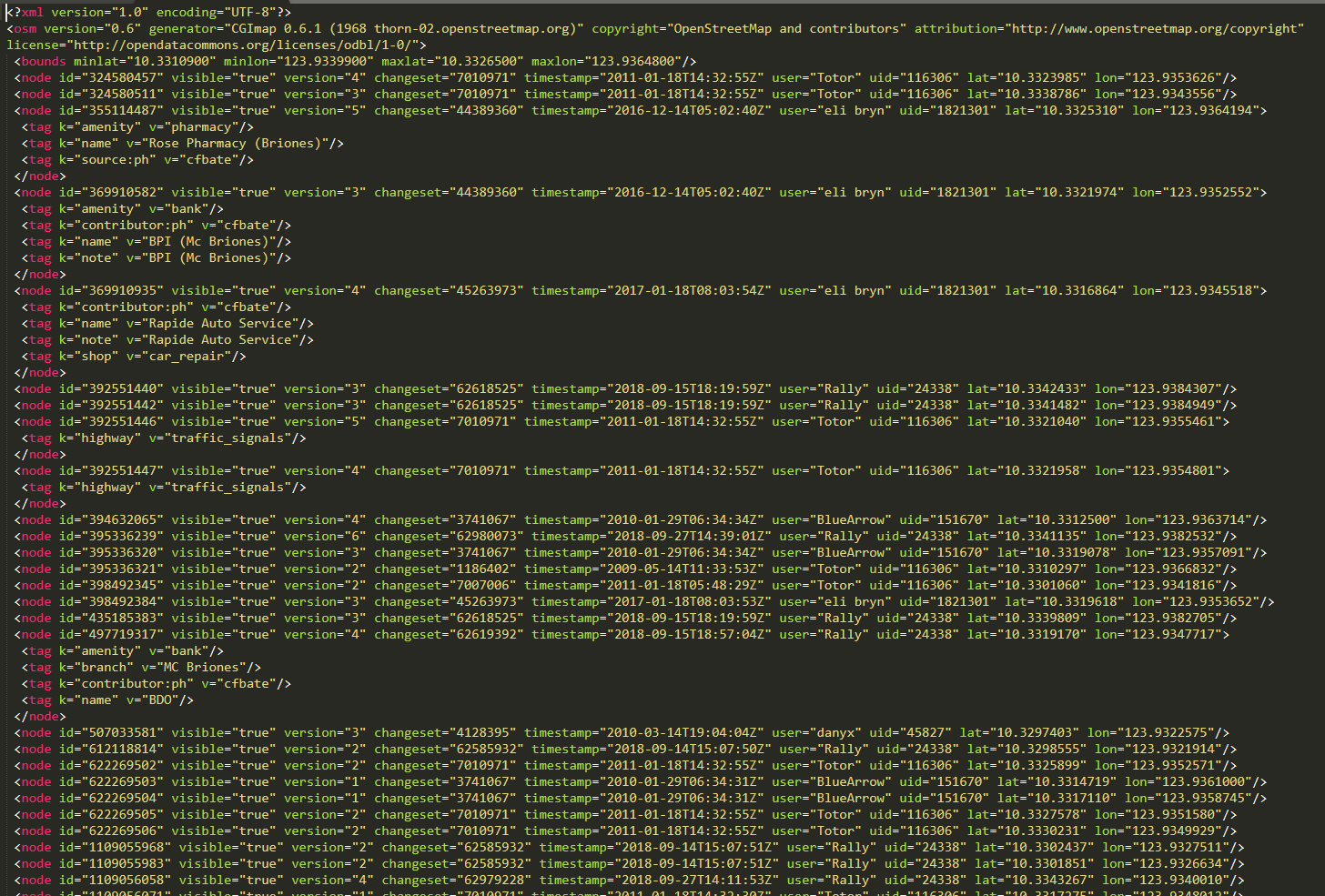
Since the developed system is a simulator, it will be retrieving data at a more unique manner. The data retrieval process of the simulator is made done during the ongoing run of the simulator by the user. The simulator will be retrieving the following data from the user which is: location, vehicle type, vehicle speed, lane of vehicle, and direction of vehicle. The simulator will be retrieving all these variables as soon as the user drops them over to the simulator.

Below are screenshots of data being collected during a simulation run.

**Data Preprocessing**

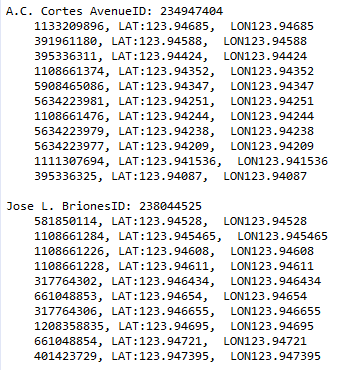
The initial process to be made done to developing the simulator would be to grab the data from the OSM API. These data are composed of the node types: junction and traffic light/s, and road types: motorway, primary, secondary, and tertiary.

Below is an image of the actual OSM XML file that is to be parsed and to be made use by the researcher to build road map and access OSM roadType and nodeType data.



*Figure 25. OSM XML file*

In building the actual road map using OSM API, the researcher will be making use of the road node coordinates. Below is a screenshot of the data collected to build a road for simulation.



*Figure 26. Parsed XML data*

Now, this part of the pre-process phase is very crucial to the simulator because most of what is needed to run the simulator would be basing off the OSM API, and to properly assess and make use of the API, the researcher needs to be thorough on what variables to use and what he can make use. For example, one of the node types of the API would be the location of traffic light/s. Now, this is very important since the simulator would then now be able to project a simulation of a traffic at a location with a traffic light and plus with the road type variable then the researcher would be able to discern the type of traffic that would normally undergo at that particular road. Hence, the labelling of the road types are as follows: ***primary*** being for the busiest routes and main roads being used, same with the ***secondary*** roads which are also labelled to be routes of similar busyness with the primary road type which means that these two are the urban road areas, the ***tertiary*** would be the road areas outside or beyond that of urban areas, which means that these are roads that would have lesser traffic, same goes for road type labelled ***unclassified***, which would mean that these roads are way off the attention of traffic or just that the traffic is not so busy and that the roads may be quieter than its other road types.

The data that would be collected by the simulator is a product of the user’s initial human interaction. The user will be required to input those variables to the simulator for it to function, and as such as it serves for its retrieval of data it at the starting point is that beginning of the processing of those data, mainly saving them to the database for it to be handled by the back-end component of the system. As the user will input the location for the simulation, then the system will begin to save the data for the whole simulation run, unless if it’s being ended than the database filled with simulation data will be refreshed. So after the input of locations, the user would now be presented with the maps, and then the user will have to add vehicles to input vehicle/s to run over the system, the vehicle/s would require an input of width, height, type and speed which are all important components to building a vehicle object. As those attributes are being saved to the database, next to be done by the user is to drag n’ drop the vehicle/s to the simulator, and as they begin to “run” the traffic simulation, the user is or may not be required to input road regulations and at the same time enable or disable traffic lights.

Now, as all of the user required actions are done, the system will now be implementing the LOS algorithm which is the initial running formula for producing the final output. The LOS algorithm will be making use of the speed, volume, and capacity of the vehicle and lane. The LOS is a major deciding factor to act as a basis to which lane should be given the right of way or to proceed to its destination. The traffic light and traffic timers will also be basing at the LOS interpreted output because the proper timing and transition of those two instruments is crucial to providing a less congested traffic situation.

**LOS Algorithm**

The LOS algorithm is an important factor to developing the formula for producing an automated traffic light and time allocator. The LOS will be the balancing scale to which the Right Of Way algorithm will be basing its outputs and decision making to which particular lane will be given right to move to its designated direction and how much time will be given for it to pass over with respect to awaiting lanes and future lanes that will be occupied with new and other vehicles.

The back-end component will be handling the computations and based on the output it will discern to what LOS category Lane X belongs to, it may be from LOS A-F.

The basis to how this formula was formulated was derived from the Signal Timing Manual and the Intersection Capacity Utilization formula. The Signal Timing Manual stated that volume is one of the key components to arriving at a decision to which lane is allowed to pass through, made to go along the human knowledge, this was the right lane of thinking. But as the researcher saw that the speed of cars is also a pressing matter to look into with respect to road and traffic systems, it was not made done by certain traffic enforcers to keep track of the speed that were being brought out by the drivers of the vehicles. Along with that input was the Intersection Capacity Utilization, the researcher saw from this tool that the capacity of an intersection, road, or lane are also pressing components that should be made noticed to how the traffic situation should be handled. So with that information, the formula was formulated to also look into effect the capacity of the lane and how many volume of vehicles can it hold up.

Here is the code snippet of the implementation of the formula to the simulator system.

**Intersection Capacity**

The intersection capacity is the formula used to determine the possible capacity that vehicle/s can occupy an intersection and also a lane.

Below is the formula for computing the capacity of an intersection.

Here is the code snippet of the implementation of the formula to the simulator system.

**Green Split Calculations**

This is the formula used for determining the length of time that is available for green signal indications on a traffic cycle run. This is an important factor to creating the dynamic traffic light and time allocator because it may serve as a service threshold to the random forest classifier as it will work out the distribution of right of way to lanes.

Below is the formula for the green split calculations.

**gi=(V/s)i/S(V/s)\*GT**

Where:  
gi = The length of the green interval for phase "i" (sec)  
(V/s)i = The critical flow ratio for phase "i"  
GT = The available green time for the cycle (sec)

Here is the code snippet of the implementation of the formula to the simulator system.

**Intersection Clearance Time**

The intersection clearance time algorithm is an important computational feature on the traffic simulator system that computes the amount of time given to a certain lane to pass over or move along its respective destination. This will make use of the already computed LOS per lane. The LOS is a key component for the rest of the computations that will be undergone throughout the simulation phase.

Below is the formula for computing for the clearance time to pass through the intersection.

**T = (LOS + L + W)/(1.47\*Vo)**

Where:

T = Intersection clearance time (sec)

Vo = Initial velocity (mph)

L = Length of the vehicle (ft)

W = Width of the intersection (ft)

Here is the code snippet for implementing the intersection clearance time algorithm to the simulator system.

**Right Of Way Algorithm**

**Random Forest Classifier**

**TESTING PROCESS**

The testing process is comprised of.

**Performance Testing**

**Data Retrieval Performance**

**Data Preprocessing and processing performance**

**Accuracy Testing**

**CHAPTER IV**

**SUMMARY, CONCLUSION, AND RECOMMENDATION**

**SUMMARY OF FINDINGS**

**CONCLUSION**

**RECOMMENDATION**

**FUTURE WORKS**

**BIBLIOGRAPHY**

**INTERNET SOURCES**

1. Sokolowski, John A., and Catherine M. Banks, eds. Principles of modeling and simulation: A multidisciplinary approach. John Wiley & Sons, 2011.
2. Barceló, Jaume. Fundamentals of traffic simulation. Vol. 145. New York: Springer, 2010.
3. Alam, J., and M. K. Pandey. "Design and analysis of a two stage traffic light system using fuzzy logic." J Inform Tech Softw Eng 5.162 (2015): 2.
4. Taha, Mohammad A., and Laheeb Ibrahim. "Traffic simulation system based on fuzzy logic." Procedia Computer Science 12 (2012): 356-360.
5. Koza, John R., et al. "Automated design of both the topology and sizing of analog electrical circuits using genetic programming." Artificial Intelligence in Design’96. Springer, Dordrecht, 1996. 151-170.
6. Petkovic, Dragutin, et al. "Setap: Software engineering teamwork assessment and prediction using machine learning." Frontiers in Education Conference (FIE), 2014 IEEE. IEEE, 2014.
7. Koonce, Peter. Traffic signal timing manual. No. FHWA-HOP-08-024. United States. Federal Highway Administration, 2008.
8. Mathew, Tom V., and KV Krishna Rao. "Introduction to Transportation engineering." *Civil Engineering–Transportation Engineering. IIT Bombay, NPTEL ONLINE, http://www. cdeep. iitb. ac. in/nptel/Civil% 20Engineering* (2006).
9. Albagul, A., M. Hrairi, and M. F. Hidayathullah. "Design and development of sensor based traffic light system." American Journal of Applied Sciences 3.3 (2006): 1745-1749.
10. Liang, Xiaoyuan, et al. "Deep reinforcement learning for traffic light control in vehicular networks." arXiv preprint arXiv:1803.11115 (2018).
11. Lämmer, Stefan, and Dirk Helbing. "Self-control of traffic lights and vehicle flows in urban road networks." Journal of Statistical Mechanics: Theory and Experiment 2008.04 (2008): P04019.
12. Gradinescu, Victor, et al. "Adaptive traffic lights using car-to-car communication." Vehicular Technology Conference, 2007. VTC2007-Spring. IEEE 65th. IEEE, 2007.
13. Ghazal, Bilal, et al. "Smart traffic light control system." Electrical, Electronics, Computer Engineering and their Applications (EECEA), 2016 Third International Conference on. IEEE, 2016.
14. Mese, John Carl, et al. "Smart traffic signal system." U.S. Patent No. 6,989,766. 24 Jan. 2006.
15. Alexander, Rob, and Tim Kelly. "Supporting systems of systems hazard analysis using multi-agent simulation." Safety science 51.1 (2013): 302-318.
16. Behrisch, Michael, et al. "Sumo–simulation of urban mobility." The Third International Conference on Advances in System Simulation (SIMUL 2011), Barcelona, Spain. Vol. 42. 2011.
17. Kamusoko, Courage, and Jonah Gamba. "Simulating urban growth using a Random Forest-Cellular Automata (RF-CA) model." ISPRS International Journal of Geo-Information 4.2 (2015): 447-470.
18. Osigbemeh, Michael, Michael Onuu, and Olumuyiwa Asaolu. "Design and development of an improved traffic light control system using hybrid lighting system." Journal of traffic and transportation engineering (English edition) 4.1 (2017): 88-95.
19. Chandler, David L. “Traffic lights: There’s a better way” (2014) MIT News Office
20. Crommelin, R W "Employing Intersection Capacity Utilization Values To Estimate Overall Level of Service", Transportation Research Board, 2010, accessed December 8, 2010.
21. Petkovic, Dragutin, et al. "Using the random forest classifier to assess and predict student learning of Software Engineering Teamwork." Frontiers in Education Conference (FIE), 2016 IEEE. IEEE, 2016.
22. Breiman, Leo. "Random Forests Leo Breiman and Adele Cutler." Random Forests-Classification Description (2015).
23. Margiotta, Richard, and Scott Washburn. Simplified Highway Capacity Calculation Method for the Highway Performance Monitoring System. No. PL-18-003. 2017.