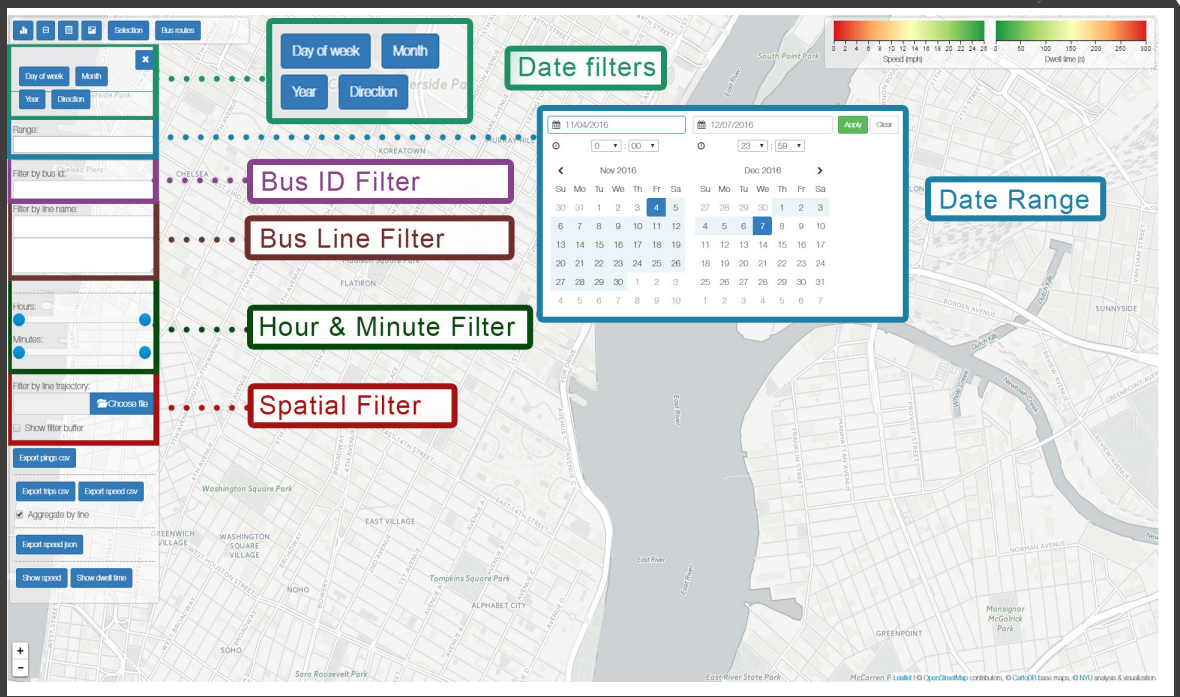
COORDINATED INTELLIGENT TRANSPORTATION SYSTEMS DEPLOYMENT IN NEW YORK CITY (CIDNY)

FINAL REPORT

TASK 8

DEVELOP DATA STORAGE AND ACCESS PLATFORM FOR MTA BUS TIME DATA



Performed by: New York University

U.S. Department of Transportation
Federal Highway Administration

NEW YORK STATE OF OPPORTUNITY. Department of Transportation

NEW YORK CITY DOT

University Transportation Research Center - Region 2

ABOUT THE PROGRAM

The FHWA, through its New York Division/New York City Metropolitan office is promoting programs pertaining to urban Intelligent Transportation Systems (ITS) in the region. The NYCDOT and NYSDOT-Region 11 Planning have taken the initiative in working with FHWA to take advantage of this FHWA program. NYCDOT and NYSDOT have developed the Training Courses and Research and Development Programs for the NYCDOT and NYSDOT Coordinated Intelligent Transportation Systems Deployment in New York City (CIDNY) which is a set of multi studies (task assignments) toward the fulfillment of the objectives of these programs.

The 2013 studies are being performed by institutions of the Region 2 University Transportation Research Center (UTRC). The studies focused on the following program areas: Construction Management, Traffic Demand Management, Dynamic Data Collection, Traffic Incident Management, Traffic Signal Timing and Detection Technologies, Strategic ITS Deployment Plan, Pedestrians and Cyclists Safety, Data Storage and Access Platform for MTA Bus Time Data.

The following tasks have been completed under this program.

- *Task 2 – Develop a multi-agency/multi modal construction management tool to enhance coordination of construction projects citywide during planning and operation phases to improve highway mobility and drivers experience*
- *Task 5 – Develop a Comprehensive Guide to Signal Timing, New Detection and Advanced Signal*
- *Task 6 – Strategic ITS Deployment Plan For New York City*
- *Task 7 – Research on Pedestrians and Cyclists Safety Using ITS Technology in NYC*
- **Task 8 – Develop Data Storage and Access Platform for MTA Bus Time Data.**

TASK 8 FINAL REPORT

UTRC-RF Project No: 57315-01-26

Project's Completion Date: January 2017

Project Title: Develop Data Storage and Access Platform for MTA Bus Time Data

Project's Website: http://www.utrc2.org/research/projects

Principal Investigator(s):

Claudio Silva
Professor
Computer Science & Engineering
Tandon School of Engineering, NYU

Kaan Ozbay, Ph.D.
Professor
Department of Civil and Urban Engineering & Center for Urban Science and Progress (CUSP)
Tandon School of Engineering, NYU

Performing Institution(s):

New York University (NYU)

TECHNICAL REPORT STANDARD TITLE PAGE

1. Report No.	2.Government Accession No.	3. Recipient's Catalog No.

4. Title and Subtitle	5. Report Date
DEVELOP DATA STORAGE AND ACCESS PLATFORM FOR MTA BUSTIME DATA	January 2017
	6. Performing Organization Code

7. Author(s)	8. Performing Organization Report No.
Claudio Silva Kaan Ozbay	

9. Performing Organization Name and Address	10. Work Unit No.
NYU Tandon School of Engineering NYCDOT 6 Metro Tech Center Brooklyn, NY 11201	11. Contract or Grant No. 57315-01-26

12. Sponsoring Agency Name and Address	13. Type of Report and Period Covered
NYCDOT 34-02 Queens Blvd. 2nd floor Long Island City, NY 11101	Final, 4/24/15-10/31/16
	14. Sponsoring Agency Code

15. Supplementary Notes

16. Abstract

Travel times can be collected from a large number of potential sources. Conventionally, fixed detectors such as inductive loops embedded in the roadway have been used to measure vehicle flows and estimate speeds. Recent technological advances and the widespread deployment of Global Positioning Systems (GPS) in consumer devices make mobile data sources a promising and potentially cost-effective way to monitor the congestion in a transportation system.

New York City Department of Transportation (NYCDOT) along with many other DOTs in the region and around the country have been using probe vehicle data for monitoring time-dependent traffic conditions and conducting before and after studies of various transportation projects. Specifically, NYCDOT has been using probe vehicle data from yellow taxis and other vehicles equipped with GPS as well as the TRANSMIT system. In this project, NYCDOT wants to automate and enhance their use of Metropolitan Transportation Authority (MTA) bus data that they are already acquiring under a protocol developed between the two agencies.

The overall goal of improving the current MTA bus data acquisition, processing, storage and querying procedures for NYCDOT comprised of 3 main tasks: The first task is to develop efficient data acquisition, storage, maintenance, querying, and visualization procedures to automate and improve the overall process of using MTA bus data. The second task is to create a web-based application that takes advantage of the MTA's on-going in- house data development efforts as well New York University (NYU) Center for Urban Science and Progress' (CUSP) extensive resources and expertise in the area of big data management. The third task is to provide recommendations to enhance the developed tool based on the experience obtained throughout this project and to incorporate this developed app and its functionalities into NYCDOT operations in a more routine manner.

In this project, we showed that it is possible to develop a simple yet powerful web based tool to acquire, store, process and visualize bus time data. This tool has an intuitive mapping user interface that can be improved by incorporating functions that can improve the robustness of the tasks at hand. The fact that the tool is web based makes it easy for the end users to access stored data and to query it without any delay or external help. Moreover, the tool enables the users to conduct a series of data visualization and analysis operations demonstrating the potential of such a web based tool for future applications.

17. Key Words	18. Distribution Statement
Bus time data, data storage and querying	

19. Security Classif (of this report)	20. Security Classif. (of this page)	21. No of Pages	22. Price
Unclassified	Unclassified	32	

Form DOT F 1700.7 (8-69)

Disclaimer

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. The contents do not necessarily reflect the official views or policies of the UTRC or the Federal Highway Administration. This report does not constitute a standard, specification or regulation. This document is disseminated under the sponsorship of the Department of Transportation, University Transportation Centers Program, in the interest of information exchange. The U.S. Government assumes no liability for the contents or use thereof.

CIDNY TASK 8:

DEVELOP DATA STORAGE AND ACCESS PLATFORM FOR MTA BUSTIME DATA

Final Report

Submitted to:

New York City Department of Transportation

January 2017

Department of Computer Science and Engineering &

Department of Civil and Urban Engineering

School of Engineering

New York University (NYU)

Table of Contents

List of Tables

List of Figures

EXECUTIVE SUMMARY

Travel times can be collected from a large number of potential sources. Conventionally, fixed detectors such as inductive loops embedded in the roadway have been used to measure vehicle flows and estimate speeds. Recent technological advances and the widespread deployment of Global Positioning Systems (GPS) in consumer devices make mobile data sources a promising and potentially cost-effective way to monitor the congestion in a transportation system.

New York City Department of Transportation (NYCDOT) along with many other DOTs in the region and around the country have been using probe vehicle data for monitoring time-dependent traffic conditions and conducting before and after studies of various transportation projects. Specifically, NYCDOT has been using probe vehicle data from yellow taxis and other vehicles equipped with GPS as well as the TRANSMIT system. In this project, NYCDOT wants to automate and enhance their use of Metropolitan Transportation Authority (MTA) bus data that they are already acquiring under a protocol developed between the two agencies.

The overall goal of improving the current MTA bus data acquisition, processing, storage and querying procedures for NYCDOT comprised of 3 main tasks: The first task is to develop efficient data acquisition, storage, maintenance, querying, and visualization procedures to automate and improve the overall process of using MTA bus data. The second task is to create a web-based application that takes advantage of the MTA's on-going in- house data development efforts as well New York University (NYU) Center for Urban Science and Progress' (CUSP) extensive resources and expertise in the area of big data management. The third task is to provide recommendations to enhance the developed tool based on the experience obtained throughout this project and to incorporate this developed app and its functionalities into NYCDOT operations in a more routine manner.

In this project, we showed that it is possible to develop a simple yet powerful web based tool to acquire, store, process and visualize bus time data. This tool has an intuitive mapping user interface that can be improved by incorporating functions that can improve the robustness of the tasks at hand. The fact that the tool is web based makes it easy for the end users to access stored data and to query it without any delay or external help. Moreover, the tool enables the users to conduct a series of data visualization and analysis operations demonstrating the potential of such a web based tool for future applications.

GENERAL INFORMATION

NYCDOT along with many other DOTs in the region and around the country have been using probe vehicle data for monitoring time-dependent traffic conditions and conducting before and after studies of various transportation projects. Specifically, NYCDOT has been using probe vehicle data from yellow taxis and other vehicles equipped with GPS as well as the TRANSMIT system. In this project, NYCDOT wants to automate and enhance their use of MTA bus data that they are already acquiring under a protocol developed between the two agencies.

The MTA is also one of CUSP partnering institutions, and over the last year, the two institutions have worked towards developing a close partnership. During these interactions, it has become clear that NYCT has already developed a "system to collect and store historic Bus Time data, while processing and validating the data against scheduled service". The effort in this project includes the development of a software tool to provide NYCDOT with seamless access to MTA Bus Time data. Some of the team members have been working with MTA and NYCDOT for several years on various data related projects. For example, MTA and CUSP staff members and researchers have organized multiple meetings and workshops covering different parts of the MTA system and its data. The research team has taken advantage of the existing partnerships and familiarity with MTA bus data to obtain the best outcome in this project.

1.1 Purpose

The overall goal of improving the current MTA bus data acquisition, processing, storage and querying procedures for NYCDOT will be achieved by completing the necessary tasks needed to achieve the following objectives:

Objective 1. Develop efficient data acquisition, storage, maintenance, querying, and visualization procedures to automate and improve the overall process of using MTA bus data.

Objective 2. Create a web-based application that takes advantage of the MTA's on-going in- house data development efforts as well NYU CUSP's extensive resources and expertise in the area of big data management. This web-based application would allow users access to the MTA data and include functionalities to create customized reports that can be used for planning and eventually real-time or near real-time travel time estimation or congestion management projects.

To achieve this objective we took advantage of the various features and components of a powerful visualization tool developed by Professor Silva and his students for NY City taxi data.

Objective 3. Provide recommendations to enhance the developed tool based on the experience obtained throughout this project and to incorporate this developed app and its functionalities into NYCDOT operations in a more routine manner.

AGENCY INTERVIEWS

2.1 MTA

2.1.1 Current Practice at MTA

MTA BusTime data is used mainly for planning purposes. A web-based interface based on in-house data including BusTime data is used for analyses and generating reports. Data is not processed in real-time. Reports are generated for the next day using filtered daily records. Point-to-point travel time and speed information can be gathered from the tool for project impact assessment.

MTA mainly uses BusTime data for planning purposes within the agency. Web-based ORGA system is an in-house developed system that uses only MTA's own data such as garage and vehicle operations, personnel data. BusTime data is a part of the tool. The system is not designed to generate reports in real-time using live feed. Historical BusTime data are usually analyzed one day after the data are collected. The results are compiled and reported using previous day records around 11 am every morning after filtering and processing of the data is completed.

Several performance measures can be extracted from the tool. These include headway analysis, speed analysis, personnel timekeeping, identifying other service problems, etc.

Additional bus data collected in MTA buses, such as Intelligent Vehicle Network (IVN) data, holds rich information that can support performance analysis however these data is found to be difficult to process and is not included in in-house tools in MTA.

2.1.2 Improvements to Existing Capabilities

Desirable features by MTA include work zone location display and resulting traffic delays information. Projects such as Transit Signal Priority could be evaluated using MTA BusTime data, with speed and bus volume information. Since entire bus fleet is equipped with GPS devices that transmit real-time location information with 30 seconds frequency, travel time analysis could be undertaken along bus routes.

2.2 NYCDOT

2.2.1 Current Practice at NYCDOT

NYCDOT does not have an accessible, easy to use web based tool utilizing MTA BusTime data feed. NYCDOT uses archived Bus Time data for project planning and evaluation by employing PostgreSQL database and Python functions to clean, store,

read/write and analyze data. This requires a certain level of familiarity with aforementioned tools to extract data.

Research team met with MTA Data Analytics team on 11/06/2015 to discuss MTA's existing in-house MTA BusTime data usage capabilities and their future needs. The summary of the meeting is below.

- MTA mainly uses BusTime data for planning purposes.
    - Web-based ORGA system is an in-house developed system that relies entirely on MTA's own data. Bus data is a part of the tool.
    - The system is not designed to use and report using real-time data. Historical BusTime data is analyzed usually the day after the data collected. The results are finalized and reported using previous day records around 11 am every morning after filtering and processing of the data is completed.
    - Several performance measures can be extracted from the tool. These include headway analysis, speed analysis, timekeeping, identifying other service problems etc.
    - Other bus data, such as IVN data, holds rich information that can support performance analysis however these data is found to be difficult to process.

2.2.2 Improvement Suggestions

- Data Analytics Group:
    - Point-to-point travel time and average speed information.
    - Segment-based travel times.
    - Work zone and related traffic delay information.
    - Project assessment for large projects such as Transit Signal Priority (TSP).
- Signals Group:
    - The tool can be used to calibrate modeling efforts of existing conditions.
    - Currently, the group uses snapshot data, i.e. manual and automated traffic counts, for model calibration and validation purposes.
    - There is a need for reliable travel time information to supplement modeling data, not necessarily in real-time.
    - Average travel time and speed information are desired to be aggregated in 15-min or any other adjustable intervals.
    - Turn movements and turn counts for buses should be extracted from the tool.
    - Project assessment is one of the major interests. Bus ID's should be queried to identify buses with TSP equipment. SBS routes and TSP implementation areas should be provided by Signals group as a layer template.
    - Buses should be filtered according to their service status: in service, deadheading, etc. Also on revenue buses should be identified.

    - For user privileges for tool access NYCDOT system, TIMS, can be taken as an example.
- TMC Group:
    - Since they work with real-time data, they are not interested in using the tool unless real-time accuracy of the travel time information is verified.

FUNCTIONAL REQUIREMENTS

The research team held interviews with transit decision-makers to discuss desired functionalities for the web-based application. There were four main topics interested by the users which included the corridor definition and travel time queries; day, date range and time definition; output with different levels of aggregation, and the map output. The functionalities requested by frond-end users are scored depending on their plausibility on a 1-5 scale, with 5 being most likely and 1 being least likely. After this evaluation, 41 potential functionalities are identified. Out of these 41 features, 25 are scored higher than 3 labeled as critical. The list of scored requirements can be seen in the appendix.

3.1 Functional Requirements for Current Bus Data

i. Bus GPS data will be acquired from MTA API developer sources in real-time and will be stored in a web-based server.
    a. The system should acquire latitude, longitude information from BusTime feed as primary input.
    b. Web-based mapping tools such as CartoDB and MapBox should be used as a primary tool for showing data points graphically on a map.
    c. The system shall use “vehicle id”, “distance along trip”, and possibly “next scheduled stop distance” fields for calculating average travel times and speeds for network links on bus routes.
    d. The system shall include the capability to calculate point-to-point average travel times using GPS readings. One main output using GPS data is the link-based travel times.
    e. The system should provide performance measures such as reliability of travel times for certain sections in the network. The tool can identify under-performing routes/links regarding running times.
    f. The data used in the system shall be stored in a web-based server which will be accessed by pre-registered users via a password protected log in.

ii. Bus GPS data will be checked for errors, discontinuities, and a processed database will be stored in addition to the original raw data.
    a. The output should be compiled only using adequately good GPS signals that are confirmed by a map-matching algorithm simultaneously.
    b. The system should identify differences between non-recurrent congestion conditions and bad GPS signals (i.e. due to device malfunctioning etc.).
    c. The system should identify trips that did not take place in accordance with the schedule.
    d. The system shall have a capability to automatically detect buses that have broken down in the middle of the trip. GPS recordings from these vehicles should be discarded in the final output.

e. The system should address travel time differences between “Limited”, i.e. express buses, or buses with “Not in Service” display in the raw data. Since these vehicles do not stop at every bus stop their travel time characteristics should be treated differently than regularly running buses.

f. The system should automatically and momentarily flag and remove outliers in BusTime data.

g. The system should be reliable when buses do not stop exactly at the bus stop.

h. Operational changes such as rerouting of lines should be automatically identified by the tool.

iii. The Bus data will be accessed remotely by pre-determined users through a secure web-portal.

a. The system shall be designed to allow users to evaluate the analysis results using a web-based interface.

b. The system should be securely protected and should only be accessible by users that are categorized with different privilege levels.

c. The raw dataset should not be accessible and not to be altered by end-users.

d. The system should be maintained on a previously determined schedule.

e. The system should have a functionality that users can report errors to the developers.

3.2 Functional Requirements for Archived Real-Time Bus Data

i. The system should store historical data in a convenient, secure and easy-to-access way.

ii. Storage practice should be optimized to use storage space effectively considering long-term data archiving.

iii. The system should have the capability to select desired time spans from historical data for the end users.

iv. Data saving for storage shall not interrupt regular real-time BusTime data functionalities and user access.

v. The system shall not delete or overwrite older data for storage purposes in order not to risk losing data.

vi. The system should allow user queries in a faster way than traditional structural database systems such as SQL or Oracle. The tool will be based on Hadoop map-reduce structure.

vii. The system shall make use of historical data for performance analysis for the following tasks:

a. Network performance: reliability of travel times for links on bus routes. The tool will provide visualizations for historical trends in travel times when there is a scheduled or unscheduled network disruption such as work zones, accidents, etc.

b. Bus service performance: schedule on-time analysis, incomplete trip percentages by line and time-of-day.

3.3 Functional Requirements for Additional Data Integration

i. The system should be flexible for additional data integration for traffic analysis. Supplementary data sources include but are not limited to:
    a. 511 NY for traffic events such as accidents, scheduled work zones, etc.
    b. TRANSCOM Data Fusion Engine including real-time travel time and incident feeds
    c. Social network data, such as Twitter and Waze, that provide user created content for transportation network status.
    d. Web-based real-time traffic and travel time information services such as Google Maps, Bing Maps, and Mapquest.
    e. Weather data from open data sources.

ii. The system should merge all open data feeds and provide users with options to select desired detail level from the additional data sources by a layer selection check box.

iii. The system should convert all data fields from different sources in a common format. There should not be different field names for similar measures, e.g. travel time should be coded in the same way for all datasets.

iv. Additional private historical data such as MetroCard swipe counts per line can be integrated into the system, if acquired.

3.4 Functional Requirements for Every Day Data Analytics

i. The system should have the capabilities for daily analytics including but not limited to following:
    a. Current travel time conditions on user-defined network regions or links. The region can be manually selected by click-and-dragging. Alternatively, users can select bus lines on a drop-down menu.
    b. The bus routes on selected region or line shall be highlighted with colors representing travel conditions. Average and instantaneous travel speed and travel time information should be provided both visually and with a table at side.
    c. Archived travel time and speed information should be accessible to the end-user along with real-time data. Short-term data can be a previous 24-hour period or previous week's average of the selected hour. Long-term data can be selected from monthly and yearly archived records.
    d. Service performance evaluation measures can be detected in real-time, and alerts can be shown on the map such as for cases of bus-bunching and schedule delays.

3.5 Functional Requirements for Scenario-Driven Data Analytics

i. The system should allow users to evaluate and compare different scenarios by creating desired routes and time spans using historical data.

ii. The system should provide supplementary travel time analysis for project monitoring. In particular for network disrupting situations such as work zones, the

tool shall provide predictions for travel time changes using historical trends from similar regions.

iii. The tool should have the capability to give travel time predictions for rerouting in certain locations. For that feature additional data sources, as well as archived data, such as map-based real-time travel time estimation services shall be used.

3.6 Functional Requirements for Visualization

i. The system should have a simple and easy-to-use interface.
ii. The map layer should make use of web-based graphical services such as CartoDB or MapBox.
iii. The system should show roads as colored links for representing network traffic conditions.
iv. The system should allow users to easily select the region, network links or bus routes.
v. The main screen should display data from real-time MTA BusTime feed.
vi. The system should have several data layers that gather data from additional data feeds.
vii. Bus stops should be colored. The system shall have an optional feature to color bus stops with multi-bandwidth depending on the service criteria. For example, if a bus stop is facing schedule delays the color of the stop might be tones of red depending on the severity of delay.
viii. The system should have a specific layer for 511 NY feed that shows locations of accidents, work zones, etc. This information can also be gathered from web-based map services' developer APIs.
ix. The system shall have a switch for alternative travel bounds and give travel time estimations separately.
x. The system should have a display bar for a different time of days. The range of time-of-days shall be pre-defined (e.g. morning peak 7am-10pm) or user-defined.
xi. The system should have the capability to show the underlying codes when a query is called by the user from the simple user interface.

3.7 Functional Requirements for Output and Reporting

i. The system output should be presented graphically for each query. Users shall be provided with easy-to-comprehend options such as bar charts, frequency distributions, and pie charts.
ii. The system should display historical travel times and speeds as time series with a different layer when requested by the user.
iii. The system should show query output in a simple table format in addition to graphical results.
iv. The system should provide color-coded values in a legend for graphical results.
v. The system should have the ability to create and export output in comma-separated value or MS Excel spreadsheet format for later use.
vi. If scenario analysis is used, the system should add basic scenario definitions and selected criteria in the output header.

IMPLEMENTED METHODS, PROCEDURES, AND CHALLENGES

4.1 Summary of Implemented Features and Potential Future Improvements

This section summarizes the functionalities of the tool, and short-term and long-term potential future improvements. These future improvements could not be implemented to the tool due to time and budget constraints.

4.1.1 Functionalities of the Tool

i. Bus GPS data which is available through MTA API developer sources is stored in a web-based server.
    a. The system acquires latitude, longitude information from BusTime feed as primary input.
    b. The system uses “vehicle id” and “distance along trip” fields for calculating average travel times and speeds for network links on bus routes.
    c. The system includes the capability to calculate point-to-point average travel times using GPS readings. One main output using GPS data is the link-based travel times.
    d. The output is compiled only using adequately good GPS signals that are confirmed by a map-matching algorithm simultaneously. Since the system collects real time data, it is possible to retrieve errors during the data collection process. These errors may due to the server’s response rate, local internet connection, or some other access problem experienced by the code. Thus, GPS signals resulting in extreme speeds due to data errors are removed from the output.
    f. Raw dataset is not accessible and not to be altered by end-users.
    g. The system can be maintained on a previously determined schedule.
ii. The system has a simple and easy-to-use interface.
iii. The system stores archived real-time data in a convenient, secure and easy-to-access way.
iv. Storage practice is optimized to use storage space effectively considering long-term data archiving.
v. The system allows users to easily select network links or bus routes. The selection can be manually executed by clicking on the map. Alternatively, users can select bus lines on a drop-down menu.
vi. The system has the capability to select desired time spans from historical data for the end users.

vii. The system shows roads as colored links for representing network traffic conditions.
viii. The system has the ability to create and export output in comma-separated value, portable network graphics (PNG), GeoJson, or MS Excel spreadsheet format for later use.
ix. The system provides color-coded values in a legend for graphical results.
x. The system can output aggregated CSV files containing average speed for the selected segments or nodes.
    a. For each different bus line in the selected section or
    b. For each selected link including the first and the last data point used for aggregation.
xi. The system allows filtering the data by multiple bus lines and prompts error when there is no data.
xii. The system can report the aggregated bus count data for selected segments within user specified time lengths.
xiii. Scenario comparisons can be made using the output generated by the system. Two different output files are required for the comparison.
xiv. Bus stops are colored. The system has an optional feature to color bus stops with multi-bandwidth depending on the service criteria. For example, if a bus stop is facing schedule delays the color of the stop might be tones of red depending on the severity of delay.
xv. The system shows query output in a simple table format in addition to graphical results.

4.1.2 Short Term Future Improvements

i. The data used in the system shall be stored in a web-based server which will be accessed by pre-registered users via a password protected log in.
ii. The system will display historical travel times and speeds as time series with a different layer when requested by the user.
iii. The system should have functionality that users can report errors to the developers.
iv. The system will have a display bar for a different time of days. The range of time-of-days shall be pre-defined (e.g. morning peak 7am-10pm) or user-defined.

4.1.3 Long Term Future Improvements

i. The system will provide performance measures such as reliability of travel times for certain sections in the network. The tool can identify under-performing routes/links regarding running times.
ii. Bus GPS data will be checked for errors, discontinuities, and a processed database will be stored in addition to the original raw data.
iii. The system should be flexible for additional data integration for traffic analysis
iv. The system should merge all open data feeds and provide users with options to select desired detail level from the additional data sources by a layer selection check box.

v. The system will have a specific layer for 511 NY feed that shows locations of accidents, work zones, etc. This information can also be gathered from web-based map services' developer APIs.
vi. The system will allow users to evaluate "What if?" scenarios using historical data.

4.2 Data Irregularities[1]

The daily responses from the Bus Time API, totaling up to 3 terabytes, were stored at a local server by researchers at Center for Urban Science and Progress (CUSP) for the whole year of 2015. All responses were successfully parsed and converted into one table containing only the required elements utilized by the tool. It is worth to mention that data extracted from Bus Time API may be different than the data contained in flat files used by DOT. Table 1 shows the differences between the flat file and the real-time responses received by using the Bus Time API.

Table 1**:** Differences between DOT flat file and Bus Time API data (2015)

	DOT flat file	**Bus Time API**
Source Database	Archived	Real-time
Sample Frequency	30 seconds	Limited by the server's response speed
Spatiotemporal Elements	Raw NMEA, including speed	Time and Location
Trip Elements	Route and status	Includes inferred elements such as Next Stop and Trip ID

Although the developed system tries to visualize the whole year data, some days are entirely missing. The missing data may be caused by some uncontrollable factors such as weather; some mitigation plan may be devised to identify errors in the API system.

[1] These errors described in this section were identified as part of the Capstone project conducted by CUSP students under the supervision of Professors Ozbay and Huy with close collaboration with Mr. Jeremy Safran of NYCDOT. (Zhou, et al., 2016)

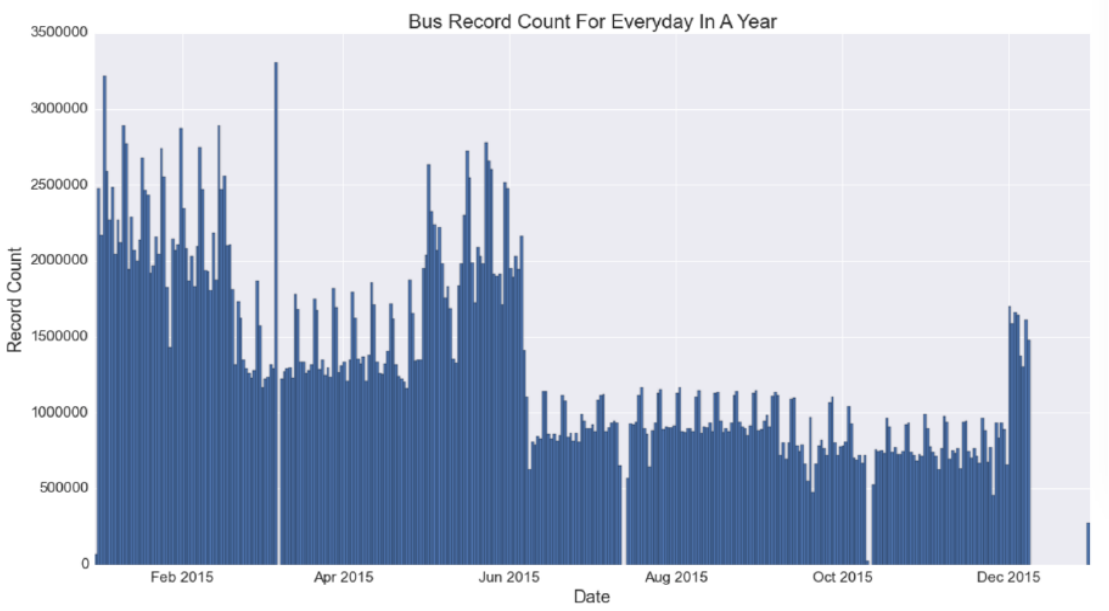


Figure 1: Visualization of Records throughout 2015 (Zhou, et al., 2016)

Figure 1 illustrates the irregularities in the data. The first data irregularity is in February 2015, second in May 2015, third in June 2015, and the last one in December 2015. It can also be seen that there are extremely high number of records on March 7^{th,} 2015 and March 8th 2015 has missing data. One can infer that data for March 7th and March 8th were merged or the date reported in the responses is not accurate.

From the data, the Global Positioning System elements of the Bus Time data appear to be transformed to adhere to the shape of the reported bus route, eliminating the lateral variations from the street. The only elements reported by the vehicle itself are the time, location and head-sign. All the other elements are inferred by the Bus Time server which is prone to errors. These inferences have to be made in real-time by the server, to remove the chance to assess the validity of them and correct the errors. It has been observed that the Bus Time server sometimes makes errors in inferring the vehicle's trip ID and such inferred ID is not persistent for the same vehicle. Trip IDs play a fundamental role in the data analysis since much of the analysis requires grouping or sorting data by using these IDs.

Buses transmit their information every 30 seconds, thus, the interval between Bus Time records is expected to be 30 seconds as long as the equipment works properly. However, the actual interval observed from the dataset collected turns out to be approximately 60 seconds. In addition, this interval even gets longer when scheduled activities increase. In other words, if more buses are transmitting their location, the server slows down to respond to the API calls. Figure 2 shows the frequency of actual intervals from the collected dataset.

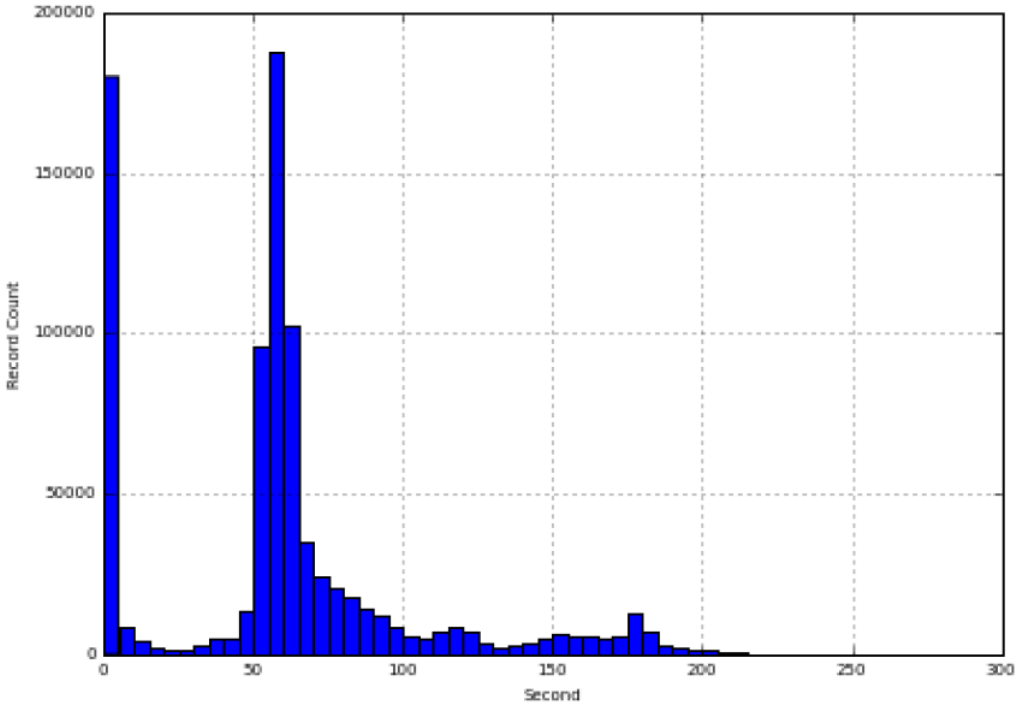


Figure 2: The Frequency of Actual Intervals from Bus Time API (Zhou, et al., 2016)

5.1 Architectural Representation

The Web Server Gateway Interface (WSGI) specifies simple rules that the server and application must conform to. It is designed to provide a high-level, ~~the~~ universal interface between Python applications and web servers. For this project, CherryPy, a minimalist Python WSGI Web Server and MongoDB, a NoSQL open-source, cross-platform document-oriented database program were used. CherryPy is a modularized component that can be utilized to execute any Python WSGI application. Whenever users make any queries on the map, the system sends the required information to the web server. The server then calls the specific application which can be filtered data extraction, aggregation or speed calculation. Using the filters defined by the users, the application will query the data server. After getting the requested results, the application will then calculate the measures and send them to the server to be presented to the user.

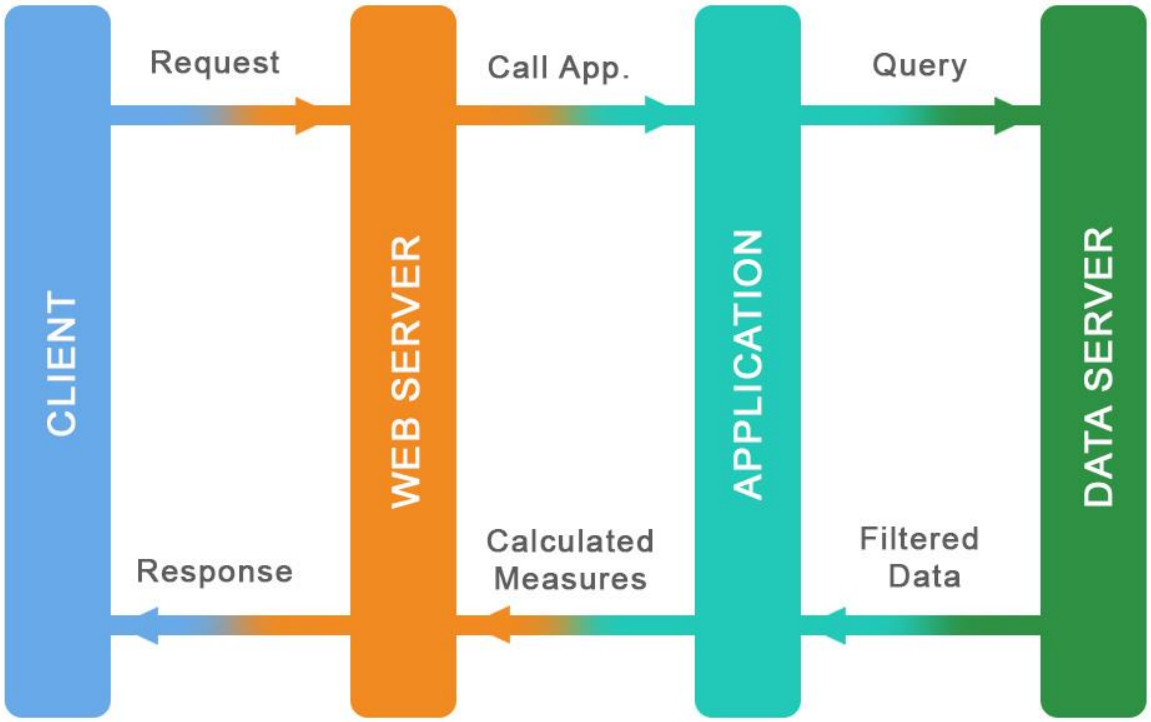


Figure 3: The System Architecture

5.2 Graphical User Interface

5.2.1. User Case View

Use-case view helps to capture the requirements so that all the stakeholders understand how the system is intended to be used. Users can either use the tool to create filters for data analysis or they can use generated data to compare scenarios. All the selection filters are explained in detail in the following sections.

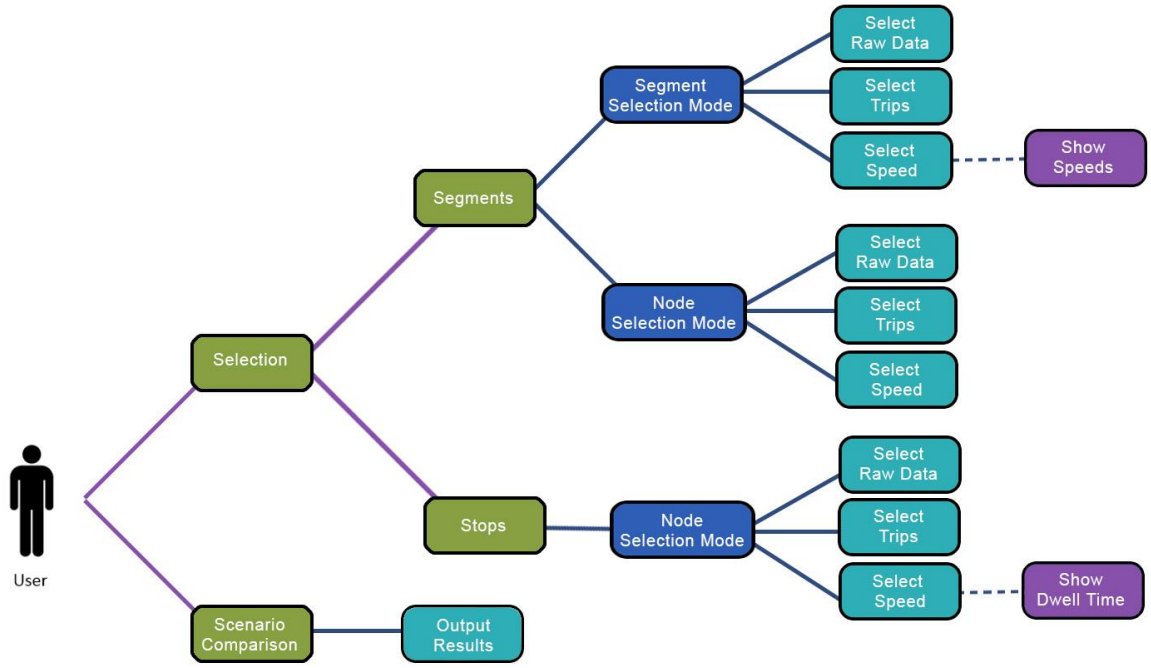


Figure 4: User Case View

5.2.2. General Query

Figure 5 shows the landing page of the system. There are six different filters that can be used to extract data. The data can be filtered by

- Day of week, month, year, and direction
- Custom data range
- Bus ID
- Bus Line (Single or Multiple)
- Time (Hour and Minute)
- Spatial filters generated by using the selection tool

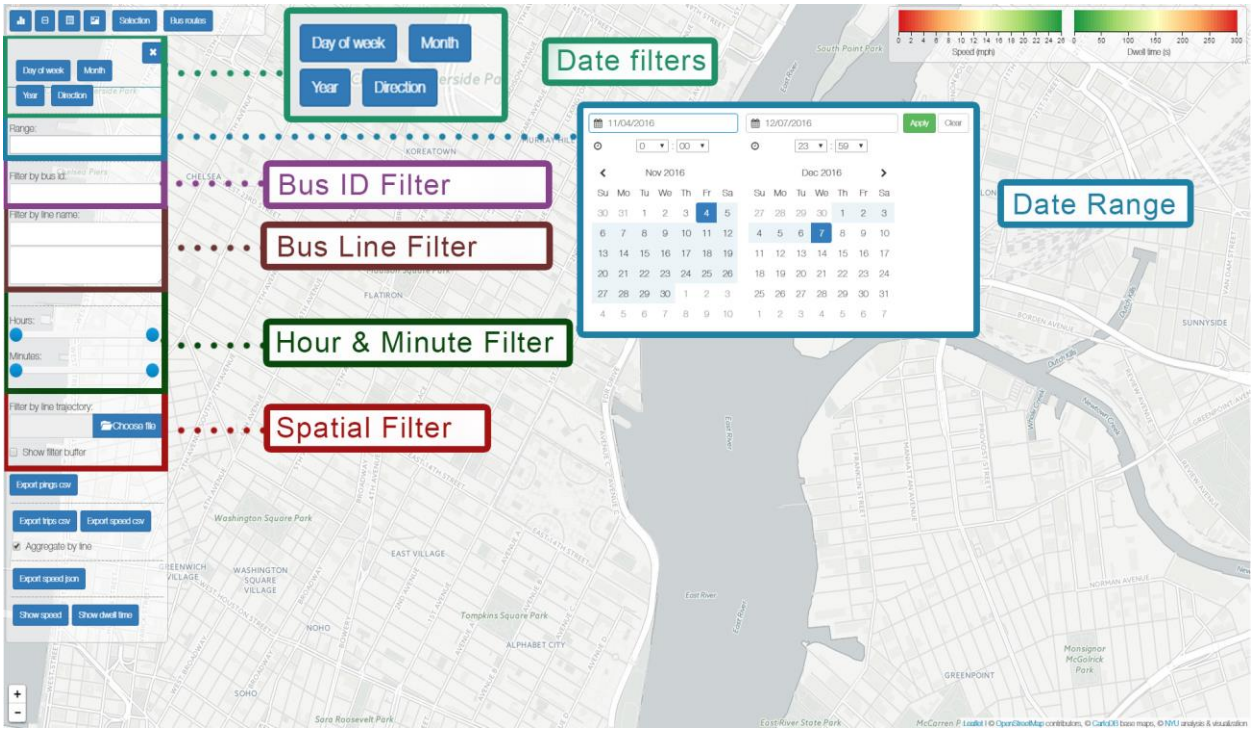


Figure 5: Data Filters

5.2.3. Spatial Selection Tool

It is also possible to filter the data by using spatial filters. The selection tool has three main modes that allow users to build a spatial filter and save it as a geoJSON file, an open standard format that describes geographical features and is supported by most GIS tools. Users can either define the start and the end point of the trips or create routes by clicking on consecutive segments. In the selection tool, the LION layer that shows all the streets in New York City will be turned on automatically

In the node selection mode, the system will try to find matching trips between the first and the following clicks. When the user clicks on the region of interest on the map, the system will snap the point to the nearest street. The direction of the clicks is not important since the travel direction can be selected using the existing filters in the system. The measures will be reported by using the data coming from all the matching trips between consecutively clicked points. In other words, users create their own segments with desired lengths by selecting origins and destinations on the map. An example selection can be seen in Figure 6.

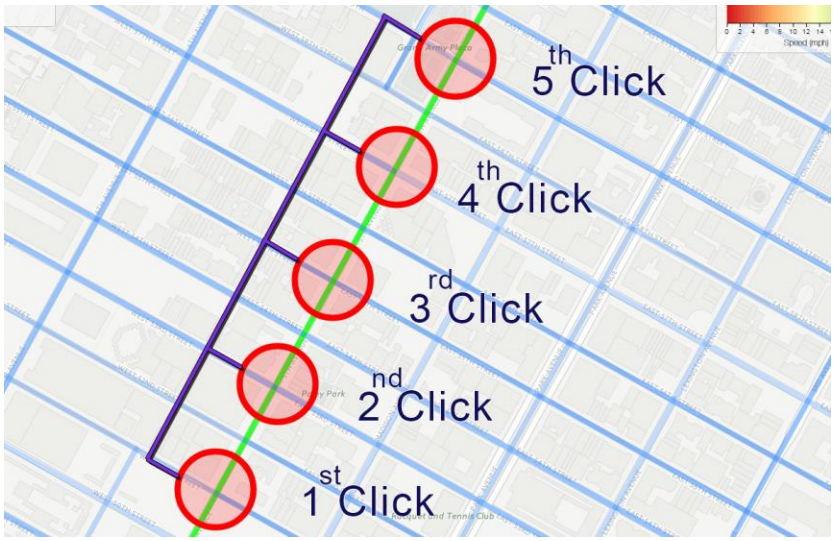


Figure 6: Node Selection Tool. The light blue segments show the loaded LION layer. The green segment shows the corridor hovered by the user when hovering the mouse. The red circles show the selected nodes.

In the segment selection mode, the segment lengths are defined by the LION layer, which is by default from intersection to intersection. This selection mode allows users to click on the street segments to build routes. When users hover over the segment to be selected, the system will change the color to green to indicate the interested corridor. Once the segment is selected, its color will turn red. Although non-continuous segments can be selected using the tool, users are encouraged to click on every segment on the desired path. Selecting non-continuous segments are useful for comparing two different routes. An example selection can be seen in Figure 7.

Figure 7: Segment Selection Tool. The green segment shows the current corridor highlighted by the user; the red segment shows selected segments.

After building the interested route on the map, users should save it as a geoJson file which can be used in the filtering process. The key difference between the two different selection methods is the way transportation measures are calculated. It is worth to mention that if the segment selection tool is used, the system will try to find multiple pings of the same bus within the segment and it will calculate transportation measures such as speed and travel time for each segment. If the node selection tool is used, the system will try to find matching records and use them to calculate the performance measures. Users can define the buffer size around the selected segments and nodes.

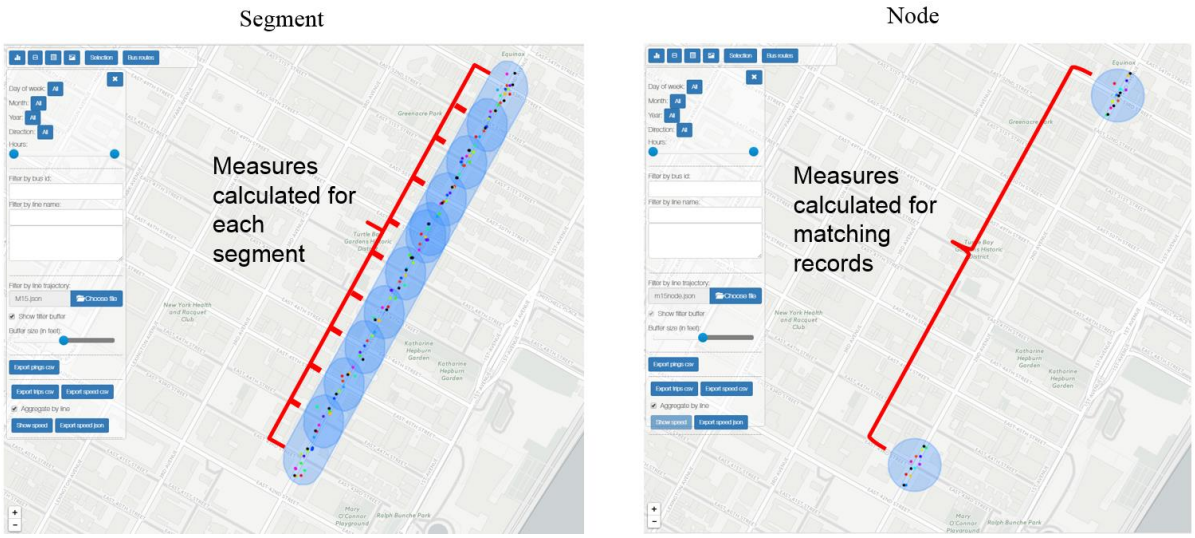


Figure 8: The Difference between Segment and Node Selection

“Stops” option in the “Selection” tool will turn on the bus stops layer. This will help users to be able to select individual bus stops using the node selection tool.

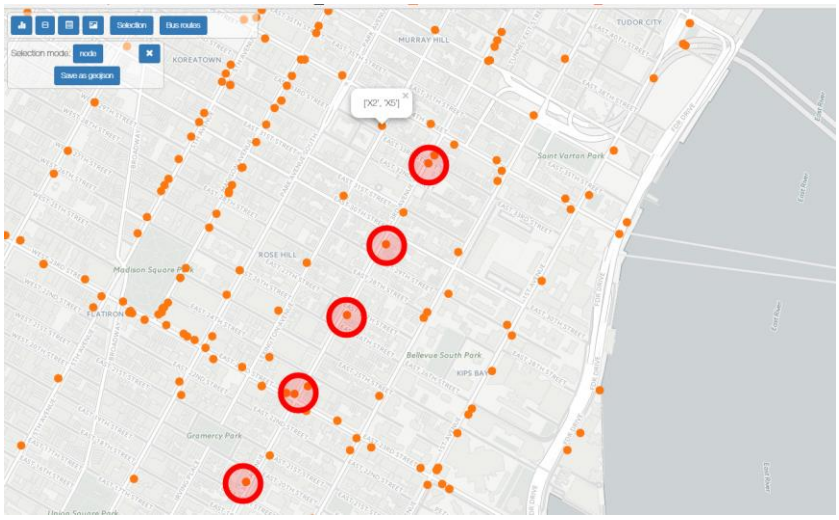


Figure 9: Bus Stop Selection. The orange circles show the location of bus stops. The red circles show the selected stops.

5.2.4. Results

Users can choose to export pings using temporal and/or spatial filters. There are four different exporting functions:

- Export pings CSV: This option will export all the pings corresponding to temporal and/or spatial filters defined by the user as a CSV file.
- Export trips CSV: This option will export all the trips corresponding to temporal and/or spatial filters defined by the user as a CSV file.
- Export speed CSV: This option will export the calculated speed values corresponding to temporal and/or spatial filters defined by the user as a CSV file.
- Export speed json: This option will export the calculated speed values corresponding to temporal and/or spatial filters defined by the user as a geoJSON file.

The user can use Microsoft Excel or LibreOffice to view the content of the CSV files. The geoJSON file can be visualized by most GIS tools.

Users have the option to aggregate trips by line while exporting. It aggregates the pings by the bus line and reports the first and the last ping of the aggregated values. If “Aggregate by line” option is unchecked, the system will report all the available data points.

The exported file name will contain information about the filters that are used. It will be a combination of the export function, day of the week, month, year, start & end hour, selected bus IDs, selected bus lines, direction and the selection mode. Figure 10 shows the header of an example exported speed file that is generated by using the segment selection mode.

speed	DayWeek.	Month.	Year.	StartHour.	EndHour.&	Ids.	Lines.	Dir.	SelectionMode.segment		
segment	line	count	mean	median	min	max	percentile	percentile75th			
0	M102	2	1.844001	1.844001	1.786276	1.901727	1.815138	1.872864			
0	all	8	4.258351	2.077931	1.027665	18.42778	1.796805	2.880651			
0	M103	2	2.290491	2.290491	2.254135	2.326847	2.272313	2.308669			

Figure 10: Exported CSV File

5.2.4.1 Results Visualization

Depending on the selection method, the system can also visualize the speed or dwell time. If the spatial filter is created via the segment selection tool, the color coded speed can be visualized on the map. When the user clicks on a segment, the system will show calculated speed values aggregated by bus line for the respective sections.

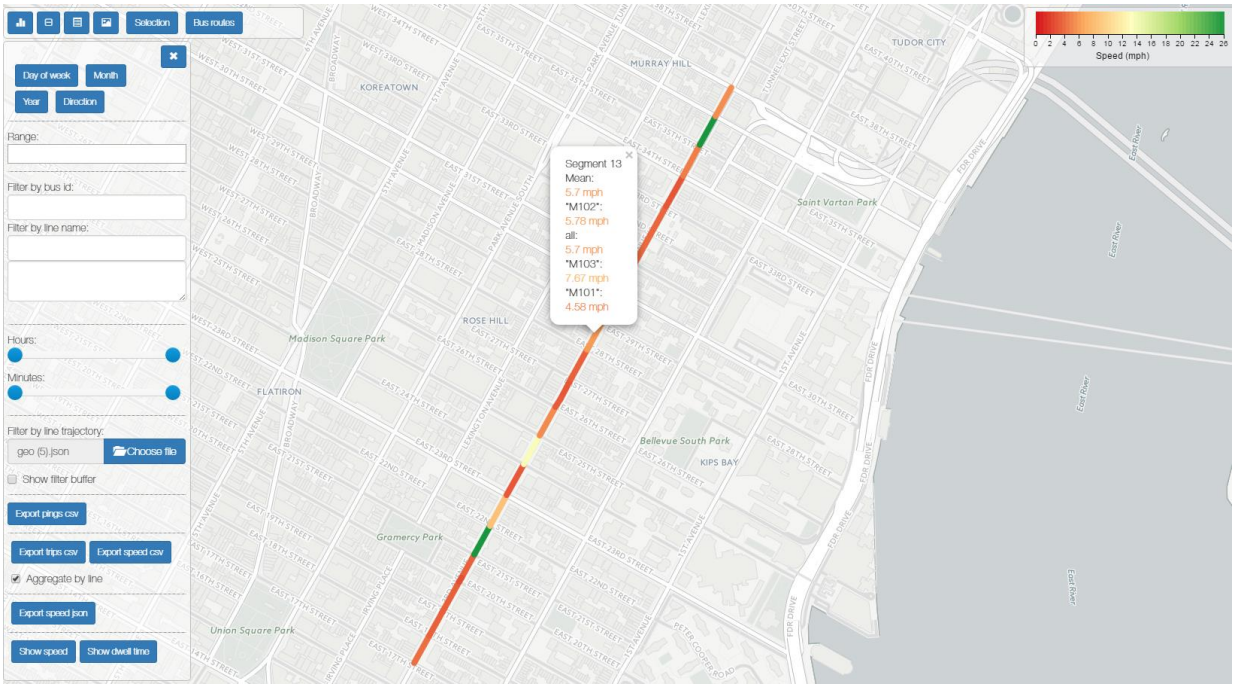


Figure 11: Segment Selection Results. The segments are color coded according to the color scale on the top right of the screen. When a user clicks a segment, more information about that particular segment is shown in a pop-up.

On the other hand, if the node selection tool is used, the system can visualize the dwell time on each node or stop.

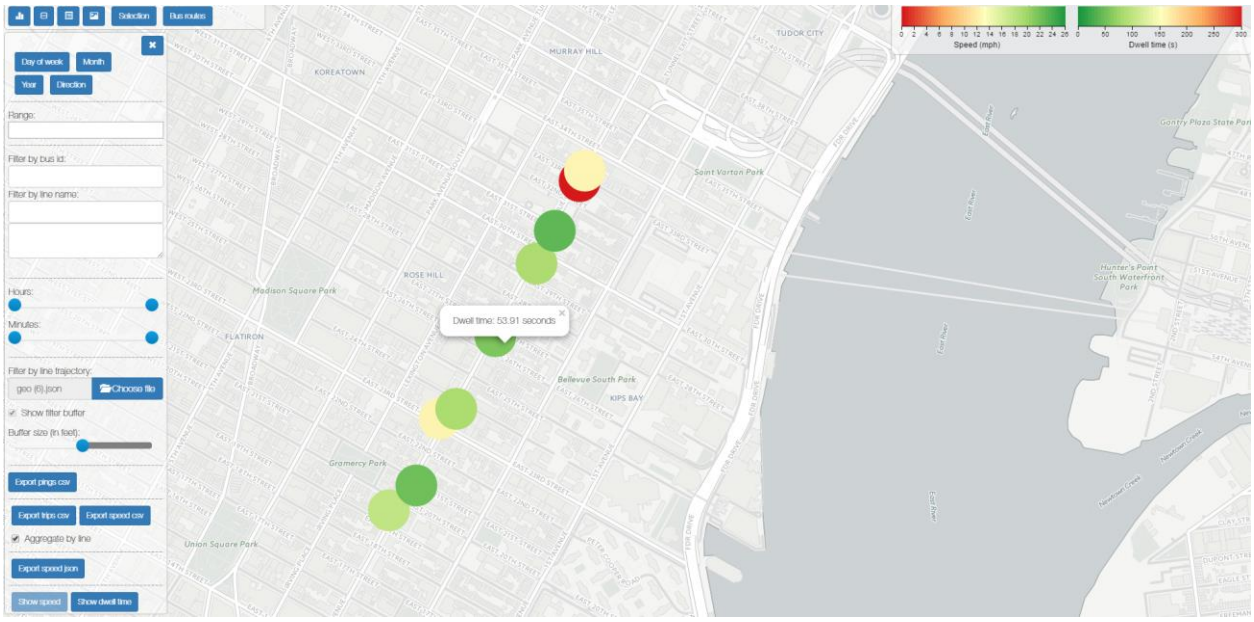


Figure 12: Node Selection Results. The circles are color coded according to the color scale on the top right of the screen. When a user clicks a node, more information about that particular node is shown in a pop-up.

5.2.4.2 Scenario Comparison

The scenario comparison tool will allow users to make before/after analysis for the desired segments. Users need to load the datasets containing exported information such as exported speed files using the tool. The system will automatically build the speed bar charts for all the bus lines for the respective segments. Users can also select individual bus lines by clicking “Line” button which shows all the available bus lines on the plot window.

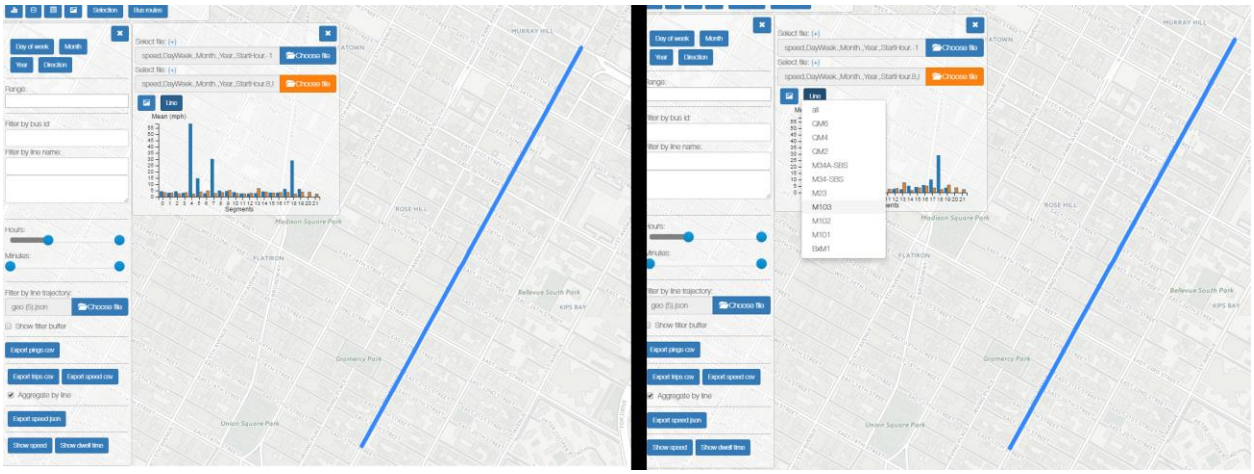


Figure 13: Scenario Comparison Tool. After loading two CSV files, the user can compare the two scenarios. The bars in the bar chart are ordered according to the clicks made by the user when creating the segment selection. If the user hovers a bar, then the corresponding segment will be highlighted on the map.

CONCLUSION & RECOMMENDATIONS

6.1 Recommendations Based on the Interviews

Table 2: Summary of Requested Functionalities

Plausibility	# of Functionalities	Completed	Not Completed
5 - (Highly Likely)	15	14	1
4 – (Likely)	10	8	2
Total	25	22	3

The NYU team scored the functionalities requested by front-end users depending on their plausibility on a 1-5 scale, with 5 being most likely and 1 being least likely. Out of 41, 25 functionalities were scored higher than 3, 9 functionalities were scored 3, and 10 requests scored lower than 3. Finally, 22 of 25 requested functionalities with scores higher than 3 were implemented in the tool. Table 2 summarizes the status of requested functionalities. 2 functionalities were not implemented due to the public API data lacking required information for deadheading buses and crashes. If requested in the future, all the other capabilities with scores lower than 3 can also be added to the tool for extended functionality. Table 3 lists the functionalities that are not completed in the final version of the tool.

Table 3: Not Completed Functionalities

Not Completed Functionalities	Status	Information
15 minute aggregated averages to match up with count data	Partly Completed	Data is aggregated for 5 minutes in the current version
Ability to query DOT flat-file to select only out-of-service or “deadheading” buses to better approximate general traffic speed.	Not Completed	Public API data do not contain required information
Display bus path through intersection after crash using bus # as ID	Not Completed	Public API data do not contain required information

While developing the tool, additional functionalities requests were made by the DOT staff. Table 4 below shows additional requests. All of these requests were implemented to the tool by the research team.

Table 4: Additional Functionality Requests

Functionality	Plausibility	Status
Users should be able to define a buffer size when selecting segments/nodes	5	Completed
Users can take a snapshot of the screen with a legend	5	Completed
Selection should have two different methodologies: Node & Segment Selection	5	Completed
Users should be able to drop nodes on the map & change the buffer size of these nodes	4	Completed

6.2 Recommendations Based on the Data

Use of the public API for real-time archiving of Bus Time data can result in an incomplete view of the data. The static file used by DOT (which is different from the public API) provides greater temporal density and additional elements that are not available through the API. Combining these two datasets would yield the ideal dataset needed for the critical analyses of traffic operations and bus time performance. If the measures such as headway, speed and travel time can be dynamically analyzed and saved for the respective links, it would make it faster to analyze these measures for different time spans. It would also not require the user to implement big data platforms, and it can be analyzed using smaller batches.

6.3 Recommendations Based on the Software Development

The developed software currently resides on a server maintained by the team due to the difficulties experienced in the implementation at the DOT servers. The research team could not grant an access to the existing MTA Bus Time portal at DOT. The developed system can be easily altered to work with a PostgreSQL database which is the current practice used by DOT. A decision has to be made for a long-term solution. The developed visualization system is compatible with various database systems and can be easily modified to work with the internal server. The tool is open source, and the source code will be shared with DOT.

One of the major challenges before and during the development of the tool was the estimation of the complexity of each task. Some tasks that were initially categorized as

trivial turned out to be more complex, and so the time and effort dedicated to each task was constantly changing throughout the project. We believe, however, that even though precious time was devoted to coding features that were ultimately scrapped or changed, this iterative process was fundamental in building an ultimately better tool.

Another additional challenge was the assumption that the users of the tool had different levels of familiarity with web tools. Because of this, we needed to dedicate more time to make sure that all possible use scenarios were simple and intuitive. On top of that, certain constraints were added to the interface to make sure that the user did not deviate too much from the intended usage scenario. Software development always has the difficult task of balancing between scope, schedule and resources. Creating an intuitive and easy-to-use tool that can satisfy the needs of users with different backgrounds is an enormous effort that goes beyond simply creating computational algorithms. It is about finding the sweet spot in a myriad of possibilities and constraints. We believe that the tool ultimately succeeds in this task and, more importantly, it provides an open and extensible platform that can be used and modified so that different metrics and processes can be tested as needed.

6.4 Conclusions

In this project, we showed that it is possible to develop a simple yet powerful web based tool to acquire, store, process and visualize bus time data. This tool has an intuitive mapping user interface that can be improved by incorporating functions that can improve the robustness of the tasks at hand. The fact that the tool is web based makes it easy for the end users to access stored data and to query it without any delay or external help. Moreover, the tool enables the users to conduct a series of data visualization and analysis operations demonstrating the potential of such a web based tool for future applications. There are few functions identified as a result of our interviews (Table 3) but not fully implemented due to data issues and /or and constraints. These should be added in the near future.

Research team will make the source code available to DOT so that potential developers can build on the developed software tool and take advantage of a dynamic open source software development environment where DOTs software developers can continue to improve the tool. Another important next step will be to incorporate data sources such as NYC taxi data and fixed traffic sensor data into this tool. Moreover, traffic information from commercial providers such as INRIX and WAZE can also be acquired and incorporated into the future versions of this tool. This will require different kind of long-term investment but such a tool with a wide variety of in-house and commercial traffic data can help NYCDOT to have a more complete and reliable picture of traffic in the City at a level that is not currently available.

Zhou, J., Cen, Y., Urbanek, M., Yuan, B., Aragno-Franco, S., Ozbay, K., et al. (2016). *BUS RELIABILITY METRICS USING PUBLIC MTA BUS TIME DATA.* New York: Department of Transportation, CUSP.

APPENDIX A

Query Example

```
database.collection.find({'$and': [

        {'VehicleLocation': {'$geoWithin': {'$geometry': {u'type': u'Polygon', u'coordinates': [[

        [-73.98822012406103,40.75074747345117],[-73.98819477785246,40.750831870923165],
        [-73.98815345353418,40.750909701921415],[-73.98809773917658,40.75097797544429],
         [-73.98802977585119,40.75103406777861],[-73.98795217535046,40.75107582332741],
         [-73.98786791981813,40.75110163744838],[-73.98778024714707,40.75111051811932],
         [-73.98769252654884,40.75110212406103],[-73.98760812907683,40.751076777852454],
         [-73.9875302980786,40.75103545353418],[-73.98746202455571,40.750979739176586],
         [-73.98740593222139,40.750911775851186],[-73.98736417667259,40.750834175350455],
         [-73.98733836255163,40.75074991981813],[-73.98732948188068,40.75066224714706],
         [-73.98733787593898,40.75057452654883],[-73.98749587593898,40.74976852654883],
         [-73.98752122214755,40.74968412907683],[-73.98756254646582,40.74960629807858],
         [-73.98761826082342,40.749538024555704],[-73.98768622414882,40.749481932221386],
         [-73.98776382464955,40.749440176672586],[-73.98784808018188,40.74941436255162],
         [-73.98793575285293,40.74940548188068],[-73.98802347345116,40.74941387593897],
         [-73.98810787092317,40.749439222147544],[-73.98818570192141,40.74948054646582],
         [-73.9882539754443,40.74953626082341],[-73.98831006777861,40.74960422414881],
         [-73.98835182332742,40.74968182464954],[-73.98837763744838,40.749766080181864],
         [-73.98838651811933,40.749853752852935],[-73.98837812406103,40.749941473451166],
         [-73.98822012406103,40.75074747345117]
        ]]}}}},

         {'hour': {'$lte': 12, '$gte': 9}},

         {'minute': {'$lte': 15, '$gte':0}},

         {'dayOfWeek': {'$in': [4]}},

         {'RecordedAtTime': {'$gte': datetime.datetime(2016, 10, 1, 0, 0)}},

         {'RecordedAtTime': {'$lte': datetime.datetime(2016, 10, 31, 23, 5)}}

]})
```

The query above retrieves bus pings between dates 2016-10-01 and 2016-10-31 in the first 15 minutes of every hour between 9 A.M. and 12 P.M. The variables in the query can be explained as follows:

- **{'$geoWithin':{}}** = Filters the data spatially by the polygons generated by the geoJson file that users create.
- **{'hour':{'$lte': 12, '$gte': 9 }}** = Filters the data by the selected hour values. lte and gte define the end and the start hour respectively.
- **{'minute':{'$lte': 15, '$gte': 0 }}** = Filters the data by the selected minute values. lte and gte define the end and the start minutes respectively.
- **{'dayOfWeek:{'$in': [4] }}** = Filters the data by the selected day of week.

- **{'RecordedAtTime': {'$gte': datetime.datetime(2016, 10, 1, 0, 0)}}** = Filters the data by the selected dates. . lte and gte define the end and the start dates respectively.

Travel times are calculated by post-processing the retrieved data. The difference between the first and the last time stamp of the matched trips are used to calculate travel times. The tool calculates the speed considering the travel time and the distance along the route, provided by the data itself.

APPENDIX B

Summary of the Interviews

DOT Bus Time Tool Front-End User Requests

Based on 2015-12-15 & 2015-12-16 Interviews at 55 Water Street and 2016-01-06 Interviews at 34-02 Queens Boulevard

Interviewees:

Hassan Adekoya & Parry Drew (IT & Telecom), Andrew Weeks (Modeling & Data Analysis), Rob Viola & Navjodh Singh (Research Implementation & Safety), Rich Carmona (Pedestrian Projects Group), James Celentano (Signals), Emad Makarious (Signals), Marvin Souza (Signals), Han Pham (Signals), Marina Ramzy (Signals), John Tipaldo (Traffic Management Center)

(#) is plausibility score on 1-5 scale, with 5 being most likely to be included in the tool and 1 being least likely

General Application Functions:

1. User Log-In and Saving System
    a. Allow users to log-in with their DOT usernames and passwords **(2)**
    b. User portal to give users access to previously saved queries and analyses **(2)**

Map-Based Corridor Selection & Attribute Specification Interface:

1. Defining your geographies of interest
    a. Ability to select segments, corresponding to blocks, to build a corridor **(5)**
        i. Custom segment sizes with grouping of blocks into a single segment for better display for longer corridors **(4)**
        ii. Allow for selection of either contiguous or non-contiguous segments in project corridor selection/querying. **(3)**
    b. Ability to select intersections as beginning and end points or for count data **(3)**
    c. Layers (toggle on and off) for SBS routes, TSP corridors, bus lanes **(4)**
    d. Ability to import GIS layer of segments to serve as querying geography **(1)**

2. Defining your period of interest

a. Ability to return results corresponding to specific days, dates, and times **(5)**
b. Ability to define custom before and after periods **(4)**
c. Seamless dual querying of both live feed data and flat file data for older records **(4)**

3. Defining your routes of interest
    a. Ability to select bus routes of interest with similar interface to busvis.org **(5)**
    b. Ability to select individual bus stops or time points **(5)**

4. Pinpointing your bus attributes
    a. Ability to select bus type that serves as a better proxy for general traffic speed **(5)**
    b. Ability to query DOT flat-file to select only out-of-service or "deadheading" buses to better approximate general traffic speed. **(4)**
    c. Ability to select and filter by bus ID # for querying only TSP-equipped buses **(5)**

5. Examining Incidents
    a. Display bus path through intersection after crash using bus # as ID **(4)**
        i. Determine if buses are complying with banned turns **(3)**

Exporting Data & Visualizations for Analysis:

1. Exporting your speed data
    a. Output travel time/speed from start of first segment to end of last segment and vice versa, indicating direction, with segments color coded by speed **(5)**
    b. Ability to export summary statistics with graphs and tables of averages as either PDF or Excel file. **(5)**
        ii. Before and after travel time bar graphs, % change, etc. **(5)**
        iii. 15 minute aggregated averages to match up with count data **(4)**
    b. Excel/CSV table output with each row corresponding to a bus trip in a given direction **(5)**
        i. Fields should include bus route, bus ID, begin time, end time, and service status **(5)**
    c. CSV flat file output with each row corresponding to a ping **(5)**

2. Exporting your bus count data
    a. Return bus volumes along a corridor, passing through an intersection, or making a particular movement through an intersection **(3)**

i. Aggregate into 15 minute periods to match up with other count data **(3)**

b. Return volume of buses stopping at a particular stop per hour, per day **(4)**

3. Exporting your maps and segments
   a. Ability to export color-coded segment speed map as either PDF, JPEG, or ArcGIS file **(4)**
   b. Ability to export segments themselves as GIS shapefile or KML/KMZ Google Earth file **(1)**

Incorporating Other Data Sources & Calculations:

1. On-time performance, reliability, and bus bunching calculations
   a. Comparison of actual bus travel time against schedule between time points **(3)**
   b. Calculate average headways at a stop **(3)**
   c. Provide measure of bus reliability based on adherence to schedule or consistency of headways **(3)**
   d. Ability to examine bus bunching at a specific time point or stop **(2)**
   e. Comparison of service attribute data to adjacent bus routes, historical data or borough-wide data **(2)**

2. Dwell time / signal delay calculations
   a. Incorporate IVN data or use stop proximity indicator in live feed to calculate approximate dwell time for individual stops **(1)**
   b. Calculate traffic/signal delay by subtracting dwell time from total stopped time **(1)**
   c. Color code stops based on dwell time similar to busvis.org **(4)**
   d. Give option of subtracting out estimated dwell times to generate better general traffic speed proxy **(3)**
   e. Incorporate per-stop ridership data from MTA model **(1)**

3. Incorporating General Traffic Speed Data Source for Comparison
   a. Allow side-by-side queries of Bus Time data with taxi GPS breadcrumb data or INRIX / HERE Maps traffic speed data. **(1)**