Middlesex University

Real Time Prediction of Road Traffic Condition in London via Twitter and Related Sources

by

Ben Sharon Rose Ben Jacob Singh

CMT4141

M00382474

Supervised by
Dr. Kai Xu

ACKNOWLEDGEMENT

Thanks for all who directly and indirectly helped me a lot in shaping my work to achieve this great extent.

A Substantial amount of support for my dissertation was given by **Dr. Kai Xu** (Senior Lecturer in VA, Dept. of Computing & Multimedia Technology, EIS, Middlesex University, London). Without his innovative and excellent supervision this work won't have been a great success, thanks a lot for the support and immense patience.

Then I thank **Dr. Ralph Moseley** (Senior Lecturer and Project Coordinator, Department of Computing and Multimedia Technology, EIS, Middlesex University, London) for his vital support as the project coordinator and his contribution in teaching me how to design the web application.

Last but not the least I would like to thank **Dr. Elke Dunker-Gassen** for teaching me the procedures on how to write a master's thesis.

.

Thanks & Regards
Ben Sharon Rose Ben Jacob Singh

ABSTRACT

Social networks have created a major impact right from its dawn with a positive growing peak. Social Networks in particular the Twitter has a huge flow of data even for every fraction of a second. Various ongoing researches are being made to mine the raw data of twitter. When it comes to the area of road traffic prediction for London numerous systems have been designed and yet many systems are on the go. The various data generated by the already existing systems include: identifying the traffic location, predicting the speed of traffic, visualizing the traffic locations on the map, updating the status on the roads etc. Also the cost incurred in developing and maintaining such complex systems on sensing traffic would be more expensive. Thus, in order to provide a cost effective system coupled with a sound insight on the road traffic conditions in London, the current system is being developed. In the proposed system the traffic updates for London provided by the TFL traffic news profile in twitter is fully utilized. Based on the gathered resource a system is designed to display the traffic condition categorized by the roads along with the latest updates made by the users on the roads. In addition to further improve and enhance the effectiveness of the system, related traffic sources such as TFL Online syndicated feeds and the Google traffic information are added. Based on the feedbacks obtained from user, the system is capable of operating as a real world application with good accuracy.

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

1.1. STATEMENT OF THE PROBLEM

At present we are in the world of rapidly advancing information technology. In one another's daily routine the information technology innovations plays a vital role. Traffic condition prediction particularly for London has been improved drastically during the last few months prior to the London Olympics 2012. But most of the application related to the traffic condition provides information about incident that has happened on the road and its location on the map. Besides this related information about the incident, the status of the road and latest traffic updates on the road by users will not be available to the users. In most of the system the incidents happening on the same road won't be clustered into one for better understanding of the traffic condition. Also the cost incurred in the overall system design is also to be taken into account. To well address these issues the following system is to be designed.

1.2. AIM

The main scope of the proposed system is to provide an effective system that shows road traffic conditions in London. In addition all recent updates by laymen and valuable sources related to the impact are made available at the same point of time. This in turn gives a sound insight on the impact and the user can decide his traffic free route.

1.3. OBJECTIVES

To achieve the scope of the proposed system, the following objectives are to be accomplished.

- Using Open-source Twitter API to retrieve data from Twitter TFL Traffic News Profile

- Retrieve data from TFL Live Traffic Disruptions by subscribing to TFL Syndicated feeds.

- To combine the traffic condition from both Twitter TFL Traffic News Profile and TFL Traffic Syndicated Feeds into a single unit.

- To display the road works in London

- Add the Google traffic information to the system
- Using Google maps API to add the Google map to the system.
- Positioning the related traffic condition markers around the main traffic condition marker in a perceivable pattern in order to avoid overlapping of Google Markers.
- To develop an attractive and interactive User Interface

1.4. THESIS OUTLINE

The following description gives a clear outline on the thesis as per the sequence of the chapters preceding the Introduction chapter.

Chapter 2 explains the evolution of the concepts for the thesis. The various design and implementation approach for the proposed system were gathered by reviewing concepts of the social network twitter along with examples on twitter visualizations, TFL Online Syndicated Feeds, Google Traffic Information, the GATI system and the traffic condition prediction in Jakarta via Twitter.

Chapter 3 introduces with the use case on the proposed system and based up on it the different requirements of the system are listed for setting up the proposed system. Also the idea gathered through the background and literature review is utilized.

Chapter 4 deals with the analysis and design required for the proposed system based on the requirements gathered. This section describes the System Architecture, UML Diagram and the User Interface Design for the system prior to implementation.

Chapter 5 explains how the system is implemented and tested based on the analysis made and the design proposed at the earlier phases.

Chapter 6 displays the demonstration on using the developed system and the evaluation made by the users of the system. The demonstration section of the thesis consists of apt screen shots to well explain how to work on the system. The evaluation section that follows includes the user feedback on the system design and its performance.

Chapter 7 involves the critical evaluation on the proposed system. To be more specific this section includes Revisiting the project scope, Review on Design and Implementation and the knowledge acquired sections.

Chapter 8 concludes the project by the usual conclusion part followed by the future work to be done on the developed system.

2. BACKGROUND AND LITERATURE REVIEW

2.1. TWITTER, THE SOCIAL NETWORK

Twitter, the data enriched online social networking site is a real-time information network that allows the user to connect to the latest information from different resources in which he/she is interested[1]. With a data limit of only 140 characters per message (in proper twitter terms called 'Tweets'), Twitter has created a great influence in the 21st century following the innovation by Jack Dorsey in March 2006. A tweet can be a photo, a video or a text conversation. Due to its popularity it is used even in business. Some organizations maintain a twitter account for updating its followers about its portfolio and growth. While some others like Transport for London are using their twitter account for their work oriented purposes such as updating the current traffic conditions. Besides the usage twitter for business, there are various useful and vital applications that can be developed by analyzing and visualizing the twitter data. Few of the examples on twitter visualizations are listed below:

2.1.1. LARGE SCALE EVENTS VISUALIZATION VIA TWITTER

Dork, Gruen, Williamson & Carpendale (2010) in their approach introduced the concept of Visual Backchannel, an interesting way to follow and to explore large scale events happening in the real world. The Visual Backchannel is composed of Topic Streams, People Spiral and Image Cloud providing various modes for visualization of large scale events. Prior to visualization, text analysis is to be performed on the raw and valuable twitter data. Text analysis involves gathering images and re-tweets, removing noisy information, removing stop words, stemming of words with similar meaning and finally associating extracted words to the real post on twitter. Once text analysis is successfully accomplished the Visual Backchannel is designed by implementing its sub-components. Finally by their working system a clear visual picture of the current happenings and current topic that is under discussion on large scale is displayed with a visualization that is easily perceivable.

[1] http://twitter.com/about

2.1.2. EMERGENCTY AWARENESS VIA TWITTER

Yin, Lampert, Cameron, Robinson & Power (2011) in their approach made use of the valuable twitter data to provide an insight into critical situations by creating an emergency awareness. Valuable data capturing for the system is carried out via twitter stream & search API and up on successful completion the data is prepared to be processed. Data processing involves Burst detection, Text classification, Online Clustering and Geo-Tagging. By burst detection an effective algorithm is used to identify the more frequently used keywords to identify unexpected incidents. Then it is followed by classification of the impact based on infrastructures. In order to collect the similar tweets related to the impact, the online clustering process is carried out. Finally the identified emergency keywords such as cyclone, earth quake, Tsunami etc. are plotted on the Google map based on their occurrence in the globe.

2.1.3. USING TWITTER TO VISUALIZE CANCER NETWORKS

Murthy, Gross & Longwell (2011) in their approach devised an innovative method to conduct a research in health sciences for identifying cancer networks via twitter. Their entire research is carried out by making an individual 'seed' user as the core to identify the cancel network focused around it. To initiate the capture of valuable data in the identified cancer network, a series of scripts were written to generate a list of friend's networks and the followers with respect to the seed. After capturing the valuable dataset from twitter, a windows-based packaged was used to visualize the identified cancer networks. Thus an innovative system that uses e-Heath for researching cancer networks via twitter is created to serve as a best mode of suggestion for the cancer patients.

The above explained examples on the visualizations using twitter well portray the importance of twitter on the valuable service they provide. From these examples it is clearly understood that the significance of twitter is not only concentrated to a particular field, instead it dominates with a widely spread span in almost all areas. In the proposed system the TFL traffic news twitter profile updates are expected to be retrieved using open-sourced Twitter REST API to get the latest traffic updates from TFL via twitter. Also to search the related traffic information from twitter the open-sourced Twitter search API is to be used.

2.2. TFL ONLINE SYNDICATED FEEDS

TFL Online Syndicated Feeds is a transport for London data service. It is generated with a policy of releasing data with a desire to provide application developers real time, high quality data coupled with its accuracy [2]. TFL has released various data feeds such as live bus arrival API, Bus stop location, Live Traffic Disruptions etc. In order to use these data feeds the developer has to subscribe the data feeds via the TFL developer's site. In the proposed system Live Traffic Disruptions is to be utilized to predict the live traffic conditions throughout London. Also to be noted that TFL Syndicated feeds are available in various file formats such as XML, KML, JSON, etc. In the proposed system the XML file format is to be selected to retrieve the valuable data.

2.3. GOOGLE TRAFFIC INFORMATION

Google Traffic Information is embedded to the Google maps to provide traffic information via a button click. It is made available by Google teaming up with the Highways Agency to provide a real-time traffic map [4]. For London, the National Traffic Control Centre (NTCC) set up at Birmingham provides the local traffic related source to Google. This traffic data is made available for the major cities in UK, US, Australia, Canada and France. To check whether the traffic information for a particular area is available or not, hover the mouse on the top right corner of the Google map. A menu with traffic layer enabled will be shown on the square widget if the traffic information is available for the given area. The level and speed of the moving traffic can be determined by the different colors used on the map to describe the traffic condition. This technology is to be added to the proposed system to further enhance the traffic prediction for the user.

2.4. JAVASCRIPT INFOVIS TOOLKIT

The JavaScript Infovis Toolkit is a JavaScript library that is used to create Interactive Data Visualizations on the webpage. It implements advanced visualizations such as Area chart, Bar chart, Pie chart, Tree Map, Space Map etc ranging from simpler to complex systems. Further enhancements are being carried out in JavaScript InfoVis Toolkit by adding the WebGL support [5]. In the proposed system the bar chart is to be implemented using the JavaScript InfoVis toolkit for showing the traffic pattern for London over a day.

2.5. ARTERIAL TRAFFIC PREDICTION SYSTEM BASED ON GOOGLE MAPS (GATI SYSTEM)

Wu, Wang & Qian (2007) in their approach designed a real-time system for predicting traffic condition for the urban streets in the Bellevue City of the Washington State. They implemented a system with the help of open-sourced web tools such as AJAX together with advanced database design and Model View Controller (MVC) application. They used the Traffic Management Centre (TMC), a centralized signal operated system to retrieve the traffic condition for the City of Bellevue. Their system architecture consists of Data Server, System Server and the User Interface. In the proposed system the similar architecture is to be used thereby replacing the costlier Data Server by already existing valuable sources such as Twitter, TFL syndicated feeds, Google Traffic Information etc.

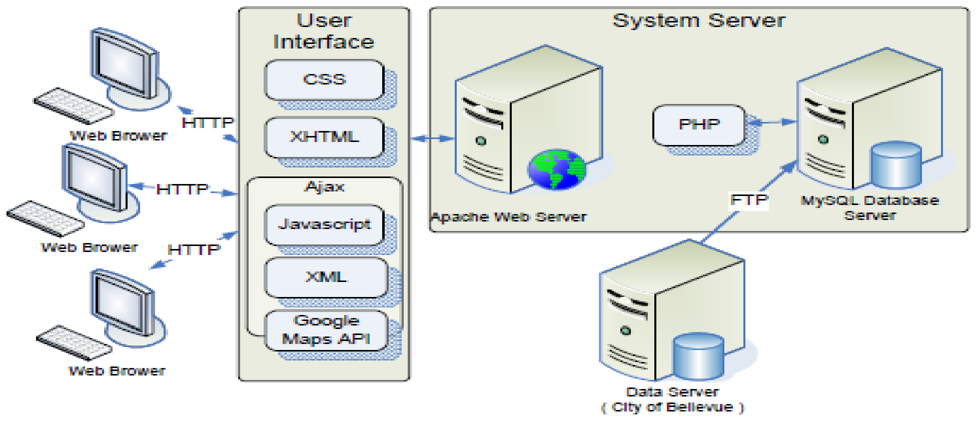


Figure 1: GATI system architecture

2.6. TRAFFIC CONDITION PREDICTION FOR ANDROID MOBILE USING TWITTER

Endarnoto, Pradipta, Nugroho & Purnama (2011) in their approach designed an Android Mobile application for determining the traffic conditions in Jakarta, Indonesia. They used a local Natural Language Processing (NLP) for processing information from a Twitter account for updating traffic condition in Jakarta. The NLP processed tweets are then tokenized and using rule based approach the parts-of-speech in the tokenized tweets are analyzed. Based on the parts-of-speech, the traffic conditions are plotted on the Google map with different colors depicting the traffic conditions. They provide only the main traffic information on the Google map and also their design is specific to their language. So in the newly proposed, the existing system scope is to be extended to provide a system that gives the user even more detailed information about the traffic condition.

3. REQUIREMENTS SPECIFICATION

3.1. USE CASE

The Use Case diagram is a generic type of UML diagram that is designed for visualizing the functional requirements of the system for the development team [9]. Since Use Case diagram is helpful for the evaluating the requirements of the proposed system, it is added prior to the listing of system requirements. The user of the system is termed as 'Actor' and the different functionalities in the system are depicted by Oval shaped use-cases. The relationship between different entities is shown by lines with an arrow head. The generic use case diagram for the web application to be designed and implemented is shown in figure 2. Before listing the requirement specifications it is better to start with the scenario under which the proposed system is to be operated. The use case diagram shown below represents best explains the scenario for the proposed system. The user interacts with the system via the user interface and it is in turn generated by a series of data processing steps.

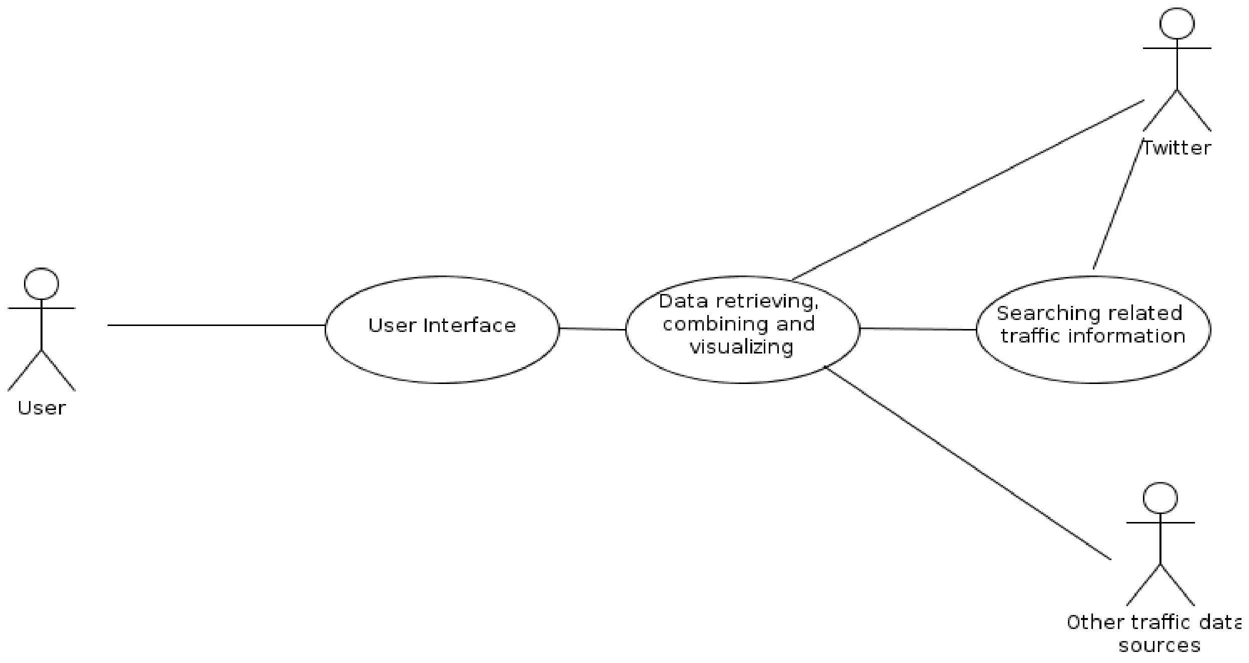

Twitter
User
User Interface
Data retrieving, combining and visualizing
Searching related traffic information
Other traffic data sources

Figure 2: Use Case diagram for proposed system

Initially the traffic data from both the twitter and related sources are retrieved and then combined based on the uniqueness. Then a search for the related traffic updates by the user is carried out on the twitter and the result obtained is appended to the corresponding main traffic data. Finally a visualization of the traffic condition is displayed on the user interface and the users are allowed to interact. So in order to design a working system to best execute in the scenario explained above the following requirements are to be met.

3.2. PROPOSED SYSTEM REQUIREMENTS

Requirement Specification is the initial and important stage in the software development life cycle. In order initiate the software development based on the scope, objectives of the system and the working scenario the different requirements of the system are listed. The different system requirements are categorized as follows:

3.2.1. SOFTWARE REQUIREMENTS

Software Requirements are nothing but the software resources and the pre-requisites that are vital to the web application. These resources are to be installed in the computer prior to run the web application.

3.2.1.1. PLATFORM

Platform is an important component when it comes to run an application. Some of the typical platforms include the Operating System (OS), libraries, programming languages etc. Operating System is the first and foremost requirement for the compatibility of a Desktop Application. Since the newly proposed system is totally web based, OS can be ignored. But Web browser compatibility issues have to be sorted out. In the proposed system HTML & JavaScript combined with Wamp Server is to be included to create a perfect platform for running the web application. In addition the required libraries are to be included to implement the desired functionality in the system.

3.2.1.2. OPEN SOURCE API

The Open Source Initiative (OSI) was invented by Eric Raymond and Bruce Perens in February 1998. It is nothing but the development based on sharing the source code thereby improving it through collaboration[2]. The Open Source API is also an OSI, developed for the sake of the developers thereby increasing the popularity and improving the quality of the source. The Open Source API's to be used in the proposed system are as follows:

3.2.1.2.1. GOOGLE MAPS JAVASCRIPT API V3

The Google Maps JavaScript API is used to add the interactive Google Map to the web page. The version 3 API of the Google Maps is designed to be faster for both the Desktop and Mobile application[3]. It is made freely available to the consumers of the API. The Google Maps JavaScript API has more built-in functions and objects that allows developers to interact with it. For instance to add a marker to the Google Map, a marker object is to be created and then the function to add the marker object to the map is to be invoked. In addition the Google Maps API has the geo-coding service built-in which is to be used to extract location from the twitter tweets and the TFL syndicated feeds.

3.2.1.2.2. YAHOO PLACE FINDER API

Yahoo Place Finder is specifically built for providing geo-coding web service to help developers aware the locations of the application's text by converting it into its equivalent geographic co-ordinates[4]. For each request, a rich set of geographic data including geo-graphic co-ordinates, WOEID and address components are returned. The WOEID is the unique identifier for identifying any geographic feature on earth, assigned by Yahoo. The WOEID returned in the response can be fed to Yahoo's Geo Planet API for further enhancement and discovery.

3.2.1.2.3. TWITTER REST API V1.1

The twitter REST API allows the developers to access the core components of twitter such as user timeline, updates on the status and other information related to

[2] http://opensource.org/history

[3] https://developers.google.com/maps/documentation/javascript/

[4] http://developer.yahoo.com/geo/placefinder/

the user. The user timeline is a collection of tweets in the chronological order[5]. Besides this the REST API provides multiple opportunities to interact with the twitter. Using REST API the developer can allow user to create tweets to the Twitter, reply to the tweets etc. In the proposed system, the REST API user timeline is to be used to retrieve the traffic tweet updates from the TFL Traffic News twitter profile.

3.2.1.2.4. TWITTER SEARCH API V1.1

The twitter SEARCH API is used to enable developer to query for twitter content. The SEARCH API includes the keyword specific twitter searches, retrieving tweets from a specific user etc. The SEARCH API has rate limits, if these limits are exceeded it may even lead to the termination of the developer's twitter account[6]. So the SEARCH API needs to be handled with an extra caution. In the proposed system the twitter SEARCH API is to be used to query twitter using the keywords extracted from the main traffic updates.

3.2.1.3. TFL ONLINE SYNDICATED FEEDS

As mentioned in the earlier sections the TFL Online Syndicated Feeds is managed by Transport for London. TFL released these valuable data about transport for the benefit of developers and for the popularity of their service. The live traffic disruption XML feed is to be requested for the proposed system to avail the traffic condition for London in real-time. The live traffic disruption data are retrieved from TFL traffic control center that monitors the incident location, its impact, nature and the time of occurrence[7].

3.2.1.4. WEB SERVER

A Web Server can either be a software or hardware that helps to avail web pages to the clients followed by a request made through the web browser[8]. In the proposed system the web server is to be used as software to avail the web application to the local network. The main intend for adding the web server to the proposed system is to access the XML file generated by the TFL Online Syndicated Feeds and to

[5] https://dev.twitter.com/docs/api/1.1
[6] https://dev.twitter.com/docs/using-search
[7] http://www.tfl.gov.uk/businessandpartners/syndication/16493.aspx
[8] http://www.webdevelopersnotes.com/basics/what_is_web_server.php

comply with W3 rules[9]. To serve this purpose a Web Server pre-configured package namely the WAMP Server is to be installed on the system. For further information on installing WAMP Server please make use of the appendix section.

3.2.1.5. WEB BROWSER

A Web Browser is an application that runs in the local operating system that is used for traversing, retrieving and presenting web content through the internet[10]. The compatibility of the web browser is a main issue while developing a web application. The proposed system is to be designed to work smooth in most widely used windows browsers. However the web application to be designed also supports other browsers, but there is a probability of few glitches to occur.

3.2.2. FUNCTIONAL REQUIREMENTS

The functional requirements are composed of technical details, manipulation of data, processing of data and other functionalities that are required to accomplish the scope of the system. The various functional requirements of the system are as follows:

- A connection to the twitter is to be established using the twitter REST API to fetch the TFL Traffic News profile updates.
- The traffic information from the TFL Online Syndicated Feeds is to be fetched by downloading XML file and parsing it using jquery AJAX.
- Combine traffic information from both the TFL Traffic News profile updates and the TFL Online Syndicated Feeds.
- Eliminate the duplicated traffic information.
- Traffic conditions for similar roads are to be clustered under one.
- The traffic information is to be geo-coded to obtain the location and these locations are to be plotted on the Google map using the Google marker.
- A search is to be made to retrieve the tweets related to the traffic information from twitter via SEARCH API.
- The retrieved user tweets are to be appended to the main traffic updates based on the roads.

[9] http://www.w3.org/
[10] http://mashable.com/follow/topics/browser/

- A pattern for plotting user tweets around the main traffic updates are to be generated and are to be plotted on the Google map.
- A timer is to be set to periodically check for the twitter updates and if there is any, the web application is to be updated.

3.2.3. NON-FUNCTIONAL REQUIREMENTS

The non-functional requirements of the system are the constraints, performance, quality and non-behavioral requirements. The different non-functional requirements of the system are listed as follows:

- A CPU (Central Processing Unit) with a processing speed of 2GHZ and 1GB of Random Access Memory (RAM) is required to execute the web server and the web application.
- JavaScript library path must be configured to run via the web server.
- JavaScript timer is required to be set up in order to initiate a function call when the execution is an asynchronous loop thereby reducing back log.
- Unnecessary looping is to be replaced with suitable statement so that it doesn’t affect the overall performance of the system.
- Proper Caching of Google map must be done so that the speed of the system is not reduced when the page is reloaded.

4. ANALYSIS AND DESIGN

4.1. SYSTEM ARCHITECTURE

The System Architecture is composed of both the Client side and the Server side, the most traditional web application architecture. Web Browser and the User Interface serves as the Client side to present the Web application to the user. The Server side includes the System Server, Data server and the Geo-coding API. The proposed design for the new system is depicted by figure 3 as follows:

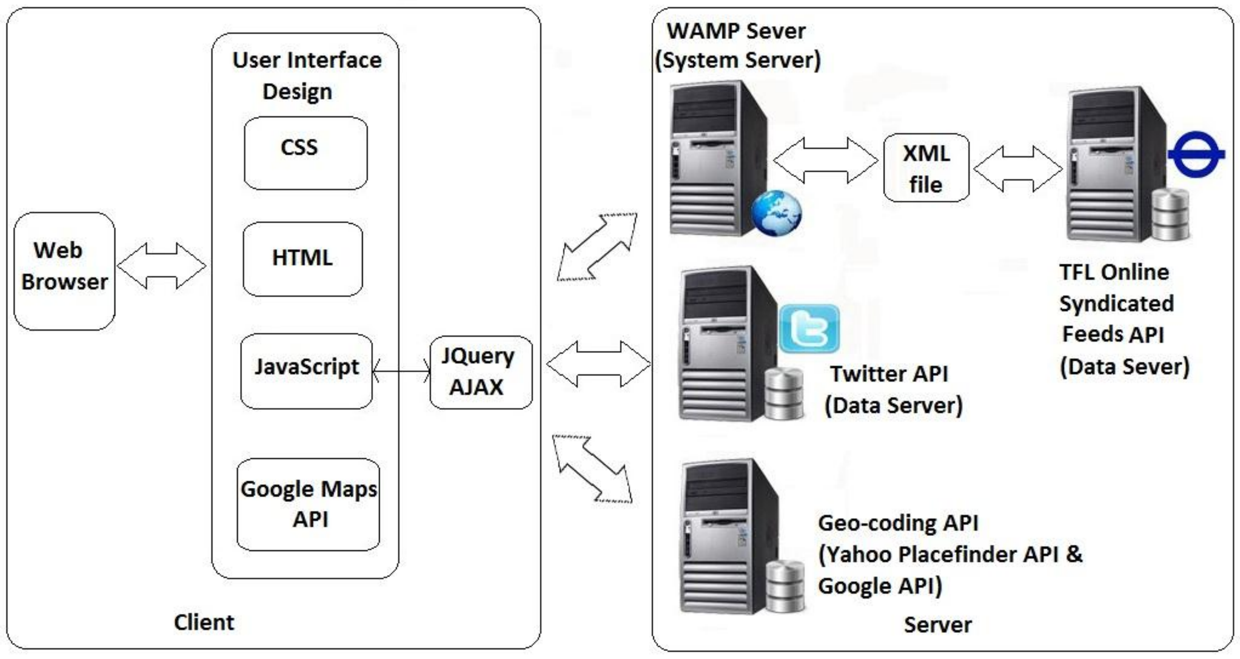

User Interface Design
CSS
HTML
JavaScript
Google Maps API
Web Browser
JQuery AJAX
Client
WAMP Sever (System Server)
XML file
TFL Online Syndicated Feeds API (Data Sever)
Twitter API (Data Server)
Geo-coding API (Yahoo Placefinder API & Google API)
Server

Figure 3: System Architecture of proposed system

4.1.1. USER INTERFACE

The User Interface is presented to the user of the system through the web browser. The web browser acts as the intermediary between the user and the web server. The User Interface for the system is developed using HTML, CSS and JavaScript programming languages. In addition the Google maps API is added to the User Interface to allow the user to interact with the Google maps. The user interface communication with the web server is carried out via the JQuery AJAX.

JQuery is a swift and concise JavaScript library designed to simplify the rigidity of HTML and to enhance AJAX interactions[11]. JQuery literally changes the way in which the JavaScript is written. The various applications of JQuery AJAX to facilitate communication with server in the system design includes Twitter REST API, Twitter SEARCH API, for parsing the XML syndicated feeds, geo-coding with Google API and Yahoo place finder API etc. To be noted that JQuery AJAX for twitter SEARCH API is asynchronous and will affect the sequential flow of the system. To avoid this JavaScript timers are employed to ensure that the flow of the program is not affected.

4.1.2. SYSTEM SERVER

The System Server is the web server for the local network and the web server used to serve this purpose is the WAMP Server. The main purpose of using the local web server is to process the XML syndicated feed from the TFL. As mentioned earlier sections JQuery AJAX technique is used to access XML feed through web server. So as per the system design if the WAMP Server is not turned on the TFL Syndicated feeds won't be available in the web application.

4.1.3. DATA SERVER

The Data Server serves as the input source for the system. The Data Server comprises of both the Twitter API and the TFL Online Syndicated Feeds API. The Twitter API in turn comprises of both the Twitter REST API and Twitter SEARCH API. The data server is not maintained locally and hence the system design is more like a cloud computing design. This in turn makes the system cost efficient and tends to improve the effectiveness of the system. The TFL Online Syndicated Feeds API similar to twitter API is maintained by Transport of London Data Service. By subscribing to the TFL data feeds the traffic information is made available freely from TFL.

4.1.4. GEO-CODING API

Geo-coding API used in the system design includes both the Google API and the Yahoo Place Finder API. To best make use of the geo-coding web services a combination of both the Google and Yahoo API is used. The system is designed in such a way that if the Yahoo API fails to geo-code the given traffic data, it is fed to

[11] http://jquery.com/

the Google API thereby increasing the number of traffic information on the Google Map. To be noted for both Geo-coding API's JQuery AJAX is used to act as intermediary link with the user interface.

4.2. UML DIAGRAM

The Unified Modeling Language (UML) is most widely used object oriented modeling language. It was introduced in the later part of the twentieth century (1997) by the Object Management Group (OMG). One of the main objectives of the UML is to provide a design language for the development community to design, create and build computer applications [9]. As UML is independent of the programming languages it has become a standard modeling language.

4.2.1. SEQUENCE DIAGRAM

The Sequence Diagram is a type of interaction diagram that the interactions between the objects sorted by the time sequence. According to Bell, D (2003) the Sequence Diagram has two dimensions namely the vertical and the horizontal dimensions. The message or call sequences sorted by the time in which they occur is shown via vertical dimensions. The target object instances to which the messages are sent is illustrated by the horizontal dimensions.

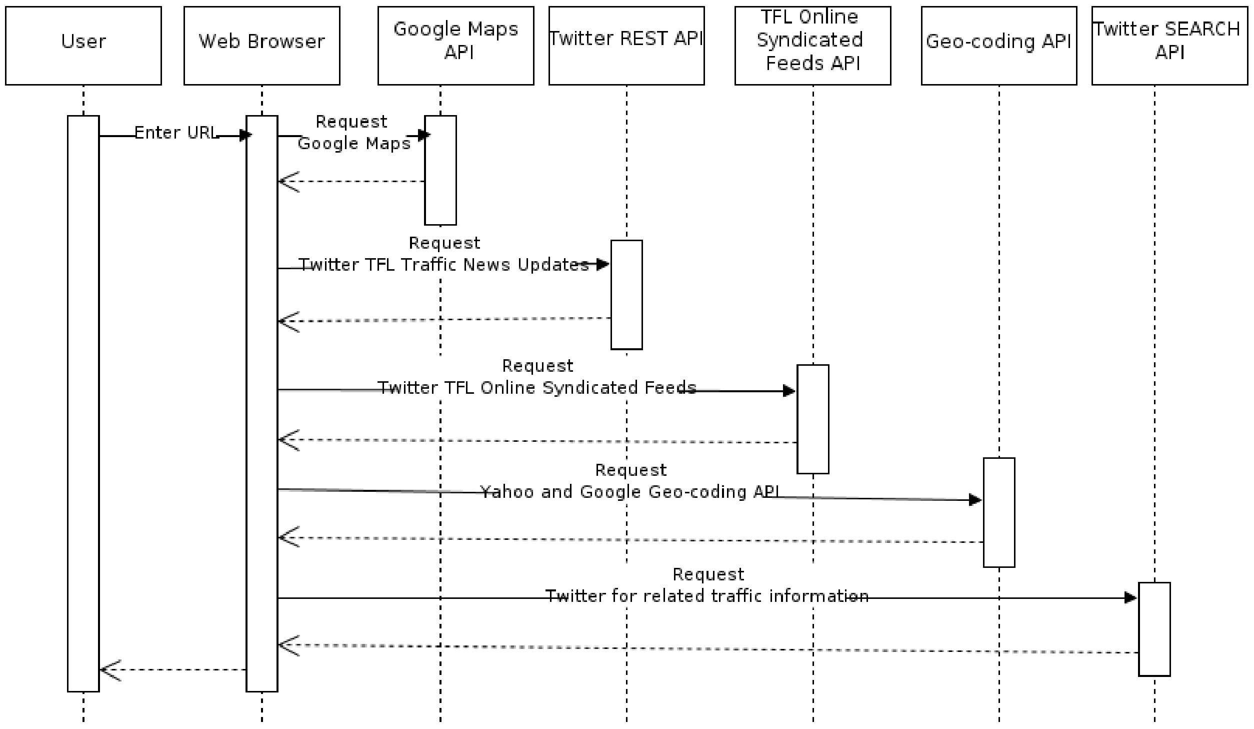
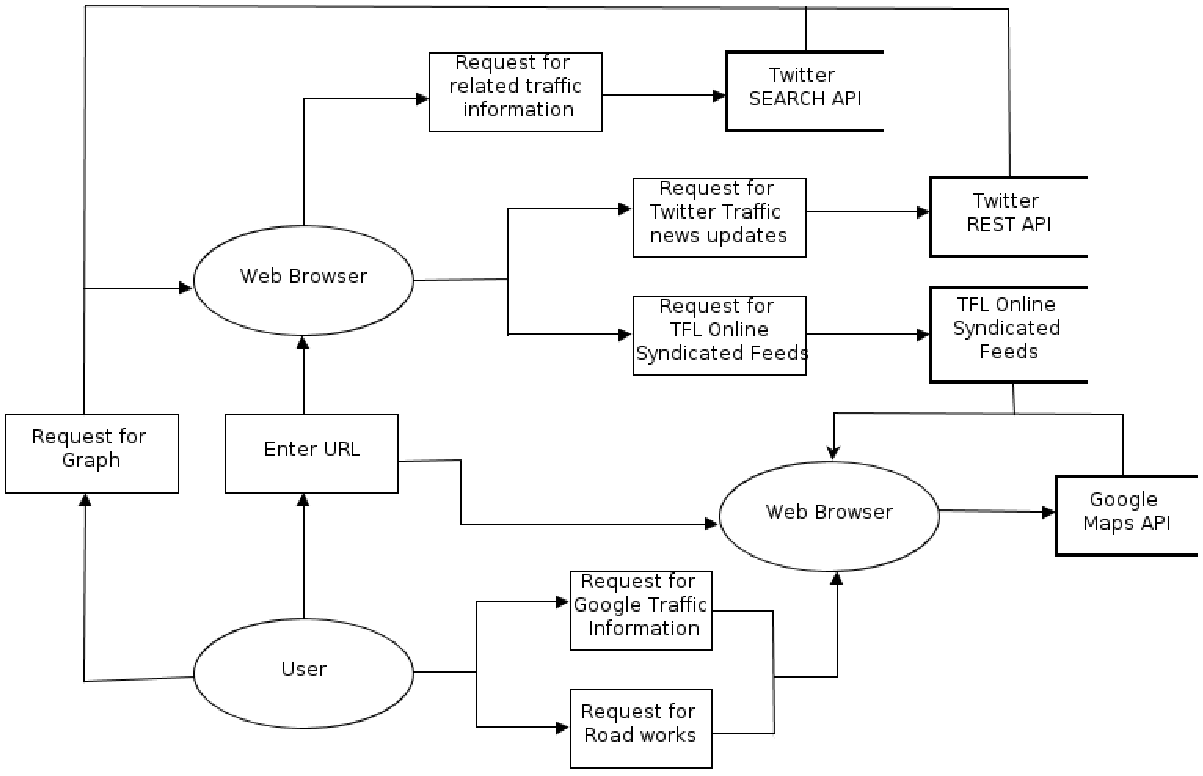

User
Web Browser
Google Maps API
Twitter REST API
TFL Online Syndicated Feeds API
Geo-coding API
Twitter SEARCH API
Enter URL
Request Google Maps
Request Twitter TFL Traffic News Updates
Request Twitter TFL Online Syndicated Feeds
Request Yahoo and Google Geo-coding API
Request Twitter for related traffic information

Figure 4: Sequence diagram for proposed system

4.3. DATA FLOW DIAGRAM

In the information system the flow of the data can be graphically represented using the Data Flow diagram. The data processing can also be visualized by using the data flow diagram. According to Siddique (2010) the DFD doesn't provide information about the sequence or timing of the processes. Also the mode of operation by the processes such as series or parallel won't be shown by DFD. However it would be useful for grasping an idea about the working of the system. The DFD for the proposed system is depicted in figure 5.



Request for related traffic information
Twitter SEARCH API
Request for Twitter Traffic news updates
Twitter REST API
Web Browser
Request for TFL Online Syndicated Feeds
TFL Online Syndicated Feeds
Request for Graph
Enter URL
Web Browser
Google Maps API
Request for Google Traffic Information
User
Request for Road works

Figure 5: Data Flow Diagram for proposed system

4.4. INTENDED FUNCTIONALITY/USER INTERFACE DESIGN

The User Interface for the system plays a vital role when it comes to the reach of an application especially for a web application. The User of the system doesn't want to acquire knowledge about the underlying architecture rather he/she prefers only the desired output. So it is important for the user interface to be effective and at the same time attractive for the users.

The User Interface Design for the proposed system consists of the Control Panel, Google maps, Information Panel and the Graph Panel. The Control Panel is composed of buttons that allows user to interact with the system. The Google maps section has the Google Map with standard interactive controls. The Information Panel is a collection of dynamically created HTML element 'div' attached to the standard HTML tables. This contains the traffic information in the chronological order as they occur categorized by roads. The Graph Panel is composed of JavaScript Infovis Toolkit pie chart to depict the traffic pattern over a day for London. Figure 5 best reveal the different layers of the user interface.

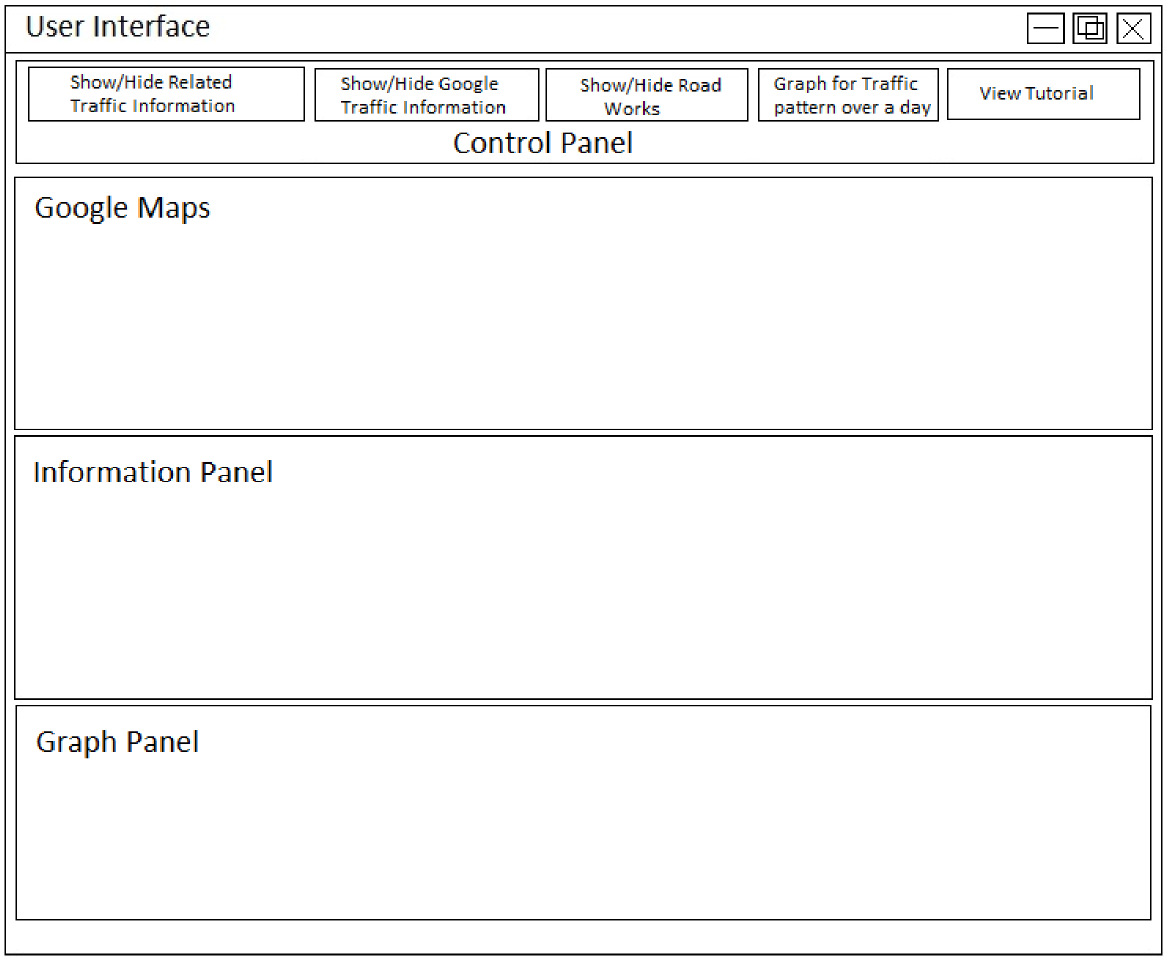

User Interface
Show/Hide Related Traffic Information
Show/Hide Google Traffic Information
Show/Hide Road Works
Graph for Traffic pattern over a day
View Tutorial
Control Panel
Google Maps
Information Panel
Graph Panel

Figure 6: User Interface Design

5. IMPLEMENTATION AND TESTING

5.1. SYSTEM PROCESS DESCRIPTION

After designing the architecture and user interface the core component of the software development life cycle is the implementation phase. The analyzed and designed system is made operational by the implementing the desired functions on the system. For the proposed system the following processes are to be incorporated to provide the traffic information for London to the user.

5.1.1. FETCH TWEETS

The twitter REST API is to be requested through JQuery AJAX by specifying the TFL Traffic News twitter profile screen name as one of the parameters. Once a request is made the response will be a JSON multi-dimensional array and it has to be parsed to obtain the required details such as tweet text, tweet time stamp, screen name, profile image etc.

5.1.2. FETCH TFL SYNDICATED FEEDS

As per the system design, both the fetching of TFL Online Syndicated Feeds and Fetching of tweets are carried out simultaneously. The TFL Syndicated Feeds API is requested for the XML file after subscribing the data feed for live traffic disruptions from TFL developer's website. Once the XML file is downloaded, it is accessed via the local web server and JQuery AJAX. It is preceded by parsing of the XML file to get the traffic information similar to the tweet.

5.1.3. BLENDING TWEETS AND SYNDICATED FEEDS

After fetching the traffic information from both sources such as the twitter and the TFL, the next process is making them into a single unit. This is achieved by storing both the data in the same array. Once the data is blended it must be sorted in the chronological order in which the incidents occur.

5.1.4. ELIMINATION OF DUPLICATED INFORMATION AND CATEGORIZATION

The sorted traffic information is then checked for duplicated information and if there is any it is removed. There is more degree of duplication as per our system design because both the input sources operate on real-time traffic conditions for London. It is followed by the Categorization of traffic information based on the roads in London. Before categorizing the traffic information it is necessary to retrieve unique texts from both source of the traffic information. This is accomplished by writing an algorithm for extracting the roads from the traffic information. The extracted roads form the unique identifier for categorizing the traffic information. If there is any occurrence of multiple incidents on the same road then they are appended under the same category. This further helps the user because he/she can get the entire traffic condition about a road under one place unlike those raw traffic updates displayed on twitter profile.

5.1.5. GEO-CODING

The categorized traffic information based on roads is then fed to the Geo-coding API via JQuery AJAX request to be geo-coded into geographic co-ordinates. As mentioned earlier for this purpose both the Yahoo Place Finder and the Google API is used. It is made possible by feeding traffic information initially to the Yahoo API and if it fails the Google API is fed with the same input. The Google API has rate limits and if in case it is fed with large sets of input the Google server stops responding to the requested IP. So to avoid this, the limit free Yahoo API is given first preference for geo-coding. The multi-dimensional JavaScript array benefit is utilized properly by storing the geo-coded co-ordinates in the same array beside the traffic information.

5.1.6. PLOTTING MAIN TRAFFIC INFORMATION

The geo-coded main traffic updates stored in the multi-dimensional twitter array is then looped and fed to the Google Maps marker one by one. The Google maps marker in turn plots the traffic information as per the geo-coded co-ordinates on the Google map. In order to distinguish the main traffic updates from the user tweets a unique color is assigned to the marker of those traffic updates. In order to enhance the look of the marker an animation for the marker is enabled by utilizing the Google maps built in functions.

5.1.7. FETCHING RELATED TWEETS

Once the main traffic updates are plotted on the Google map, the main traffic updates is prepared for querying twitter again to find related traffic information updated by different twitter users. This is practically implemented by using the familiar JQuery AJAX request to the twitter SEARCH API. The keyword for the request is the road that is extracted from the main traffic updates using the newly written algorithm. The response of the request will contain the same set of information similar to the main tweets. So the required information is extracted from them and stored in a separate array for the user tweets.

5.1.8. APPENDING USER TWEETS TO MAIN TRAFFIC UPDATES

The process of appending user tweets to main tweets is similar to the categorization of the main traffic updates. The related tweets that are fetched by querying the roads are appended under the main tweets on the same road. By implementing the system in such a precise way, the whole information of the traffic on the road will be made available to the users of the system.

5.1.9. CREATING A MARKER PATTERN

After appending the related user tweets with the main traffic updates the next step is to find a way to plot them in the Google map. The plotting of user tweets must be done in such a way that they don't overlap with main tweets. So a definite pattern to plot the user related tweets around the main traffic updates must be designed. To solve the overlapping marker issue a new algorithm for creating the markers in a successive square pattern around the main tweets was written. Then after successive testing with sample inputs it was implemented in the current system.

5.1.10.PLOTTING USER TWEETS

The geographic co-ordinates generated after implementing the newly written algorithm is fed to the Google Maps marker as usual. The color of the marker is changed to differentiate the user related tweets with the main traffic updates in the Google Map.

5.1.11.CREATING DYNAMIC DOM ELEMENT

The creation of the dynamic DOM element namely the div is done parallel after geo-coding the traffic information for the main traffic updates and after creating the pattern for the user related tweets. The dynamically created div is made structured by placing each div within the HTML table. The coding for dynamically created div is made at ease by using the simple and swift JQuery. The main intention for creating the div is to provide an information panel by which the user can read the traffic condition on the roads other than the Google Map.

5.1.12. GENERATION OF GRAPH

After setting up the information panel with traffic condition on the roads the next implementation is to provide an additional enhancement to the system. By using JavaScript Infovis Toolkit, a bar chart is selected to show the traffic condition for London over a day. To be noted that the input for the graph is generated in a separate array and at the same time of the traffic information retrieval from the API's. The JavaScript Infovis Toolkit bar chart is designed in such a way that it accepts JSON array as input. In the implementation part, the code for converting JavaScript array to JSON array is written. So when a button click event is fired the graph is generated on the run time and is dynamically made available to the user.

5.1.13. PERIODICALLY UPDATING THE WEB PAGE

Once the basic functions of the proposed system are finished the next step is to provide a way to update the traffic information displayed. This is made possible by using the JavaScript interval function to invoke methods after a given time interval to check if there is any update on the current system. In case if there is any new traffic updates available the entire Google maps marker is removed from the system and all stuffs are loaded again without getting the web page to reload.

5.1.14. ADDITIONAL FEATURES

The additional features other than Graph that are used in the proposed system include the Google traffic information and the road works in London. The Google traffic information is added to the system by adding the traffic layer object to the Google Map. The system is implemented in such a way that the Google traffic information can be shown or hidden by firing button click events. Regarding the road works in London the data is gathered from the TFL syndicated feeds at the time of parsing the XML file. The data related to road works are stored and geo-coded separately. Finally it is made available to the user via button clicks similar to the Google traffic information.

5.2. TESTING

Testing is an important phase in the software development life cycle to best evaluate the quality of the system. This is a core area to be handled with caution to ensure that the end user of the system is satisfied with the end product. In order to perform an effective testing it is vital in first hand to well digest the working of the system on the whole. To ensure that the developed web application operates without any flaws the testing strategy for the system is sub-divided into Unit Testing, Integration and finally followed by the system testing.

5.2.1. UNIT TESTING

The main core concept of the unit testing is to isolate the system in whole to smaller units. By doing so, the problems in the smaller units can be well addressed and sorted out with an ease. In an object oriented system design the unit testing can be performed by isolating the classes. Initially for the unit testing to be carried out in the functional oriented proposed system, the different functions in the system were isolated. Then for each unit i.e. the functions, a test was carried out for different range of inputs. After a positive result was obtained, the inner components of the functions such as the arrays, conditional statements, loops etc. were tested with a range of inputs. The same process was repeated for all the units in the system. Once all the tests were successfully accomplished the system was all set for the next level of testing to be carried out.

5.2.2. INTEGRATION TESTING

The Integration Testing is followed by the Unit Testing based on the traditional testing strategy. In the integration testing the modules or the smaller units that are individually tested are integrated into a single unit and then the testing is carried out on whole. By doing so the interactions between the different units can be best tested. To be noted, this is another crucial level of testing. After undergoing the unit testing for the proposed system the smaller units termed the functions were combined into a single large unit. Then the different range of inputs was fed to the system and the system was checked as a single unit. The results of the testing were analyzed one by one thereby observing the interactions between units. Once the desired results were obtained the system was made ready for the next level of testing i.e. the System Testing.

5.2.3. SYSTEM TESTING

System testing is a kind of black box testing in which the knowledge of the underlying programming code and the logic are not required. The testing is carried beyond the requirement specification of the hardware or software. The main goal of the testing is to ensure that the system responds to the user interactions and to the input data source. Based on the system testing performed the following are the test cases obtained.

5.2.3.1. SYSTEM TEST CASES

Test Case: 1

Description: Request the system to predict traffic condition by disabling the input source twitter TFL traffic news updates

Properties:

> **Input Source:** TFL Syndicated Feeds
>
> **Expected Result:** Traffic condition prediction by TFL Syndicated Feeds
>
> **Actual Result:** As expected.

Test Case: 2

Description: Request the system to predict traffic condition by disabling the input source TFL Syndicated Feeds

Properties:

> **Input Source:** Twitter TFL traffic news updates
>
> **Expected Result:** Traffic condition prediction by Twitter TFL traffic news updates
>
> **Actual Result:** As expected.

Test Case: 3

Description: Enabling both Twitter TFL traffic news updates and TFL Syndicated Feeds

Properties:

> **Input Source:** Both Twitter and TFL Syndicated Feeds
>
> **Expected Result:** Traffic condition prediction by both Twitter and TFL Syndicated Feeds
>
> **Actual Result:** As expected.

Test Case: 4

Description: Requesting the Graph by disabling Twitter TFL Traffic updates

Properties:

> **Input Source:** TFL Syndicated Feeds
>
> **Expected Result:** Graph showing traffic pattern for a day using TFL Syndicated Feeds
>
> **Actual Result:** As expected.

Test Case: 5

Description: Requesting the Graph by disabling TFL Syndicated Feeds

Properties:

> **Input Source:** Twitter TFL traffic news updates
>
> **Expected Result:** Graph showing traffic pattern for a day using Twitter TFL traffic news updates
>
> **Actual Result:** As expected.

Test Case: 6

Description: Requesting the Graph by enabling both Twitter TFL traffic news updates and TFL Syndicated Feeds

Properties:

> **Input Source:** Twitter and TFL Syndicated Feeds
>
> **Expected Result:** Graph showing traffic pattern for a day using both Twitter and TFL Syndicated Feeds
>
> **Actual Result:** As expected.

5.2.3.2. GRAPHICAL USER INTERFACE TESTING

The Graphical User Interface (GUI) Testing is a vital part of the System Testing. The users of the system rely only on the user interface to work with perfection. Initially prior to undergo the GUI Testing for the developed system, the different components of the user interface were checked for the basic functioning. Then the different GUI operations were checked through a keen observation of the sequence of step involved in the execution to attain the desired output. Once the entire GUI operations were checked the system is ready to serve the users without any flaw.

6. DEMONSTRATION AND EVALUATION

6.1. SYSTEM WALKTHROUGH

The System Walkthrough gives you comprehensive details about how the system works. The description about how to run the system combined with the screen shots gives a better perception of the working system. This serves as a better solution to best understand the system instead of setting up the environment such as configuring web server for running up the application right from scratch. The following are the different steps involved in executing the web application.

6.1.1. CONFIGURING THE WEB SERVER

Configuring the web server starts with downloading the TFL syndicated news feed and uploading it to the web server folder located in the local system server. It is followed by starting the local system server; according to the proposed system design the WAMP server is started. Now the web application is well prepared to be launched. It is to be noted prior to running the web application; the URL for the web application must include the running .html file with its complete local address starting with its root directory in the system server.

6.1.2. LOADING THE PROGRAM

The web application URL is typed on the compatible browser as discussed in the requirement specification and the application begins to load with a standard JQuery user interface library progress bar. It gets into account the JavaScript timers, the time for geo-coding and the time for fetching data from the twitter API etc. Once the progress bar completes the system is ready for the user interaction.

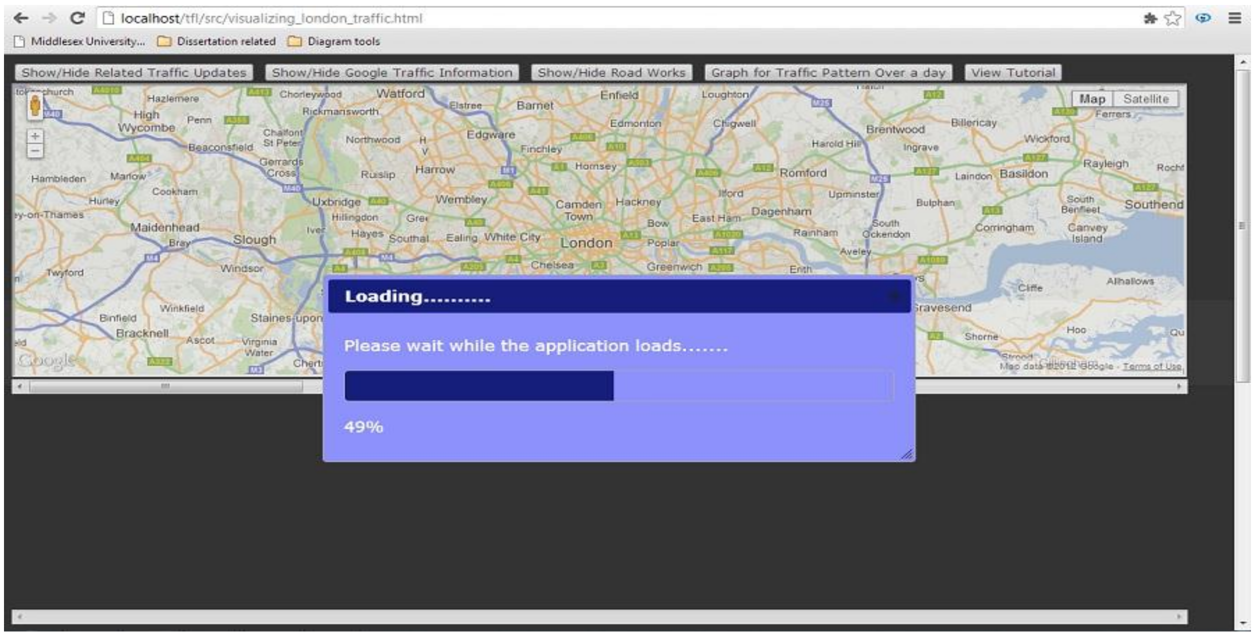

localhost/tfl/src/visualizing_london_traffic.html
Middlesex University...
Dissertation related
Diagram tools
Show/Hide Related Traffic Updates
Show/Hide Google Traffic Information
Show/Hide Road Works
Graph for Traffic Pattern Over a day
View Tutorial
Map
Satellite
Loading..........
Please wait while the application loads.......
49%

Figure 7: Loading the program via progress bar

6.1.3. DESCRIPTION OF THE GOOGLE MAPS

As mentioned in the earlier sections the Google Maps is loaded to the web page with basic interaction controls via the Google Maps API. The basic interactions available on the Google Maps include the street view pan on the left top corner of the Google Maps. Beneath the street view pan is the zoom in and zoom out pan. On the right top corner are the square widgets such as the Map and Satellite view widgets. Also the Google map is drag enabled and even responds to the other mouse events such as scrolling and more such interactions.

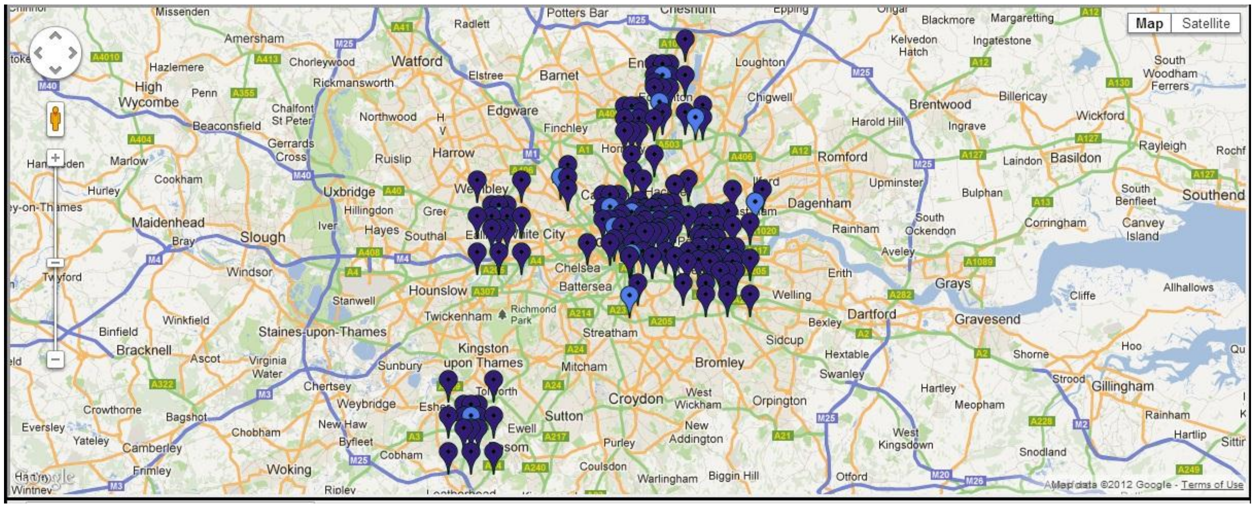


Figure 8: Google maps description

6.1.4. DYNAMICALLY CREATED HTML DOM ELEMENT

The detailed description of how the div is created and linked to the user interface in the structure of the HTML table is well covered in the implementation section of thesis. In this demonstration section the additional enhancements added to div are briefed. The dynamically created HTML Document Object Model (DOM) Element namely the div is colored in such a way that there is a clear distinction between the main traffic updates and the related tweets. To make it clearly perceivable the main traffic updates is given the light blue colored background and the related traffic tweets are given the dark blue background as depicted in figure 9.

To further enhance the tracking of the Google marker with the dynamically created div, a link is provided between the div and marker. The link is made lively by popping out an info window above the Google marker on the map with the traffic information on it. So when the cursor is hovered over the successive div on the map the info window keeps on popping out right above the marker. In addition the dynamic div is provided with an invisible scroll bar similar to the one used in the Facebook news feeds. This is made possible by adding the Rochal's slim scroll bar open source JavaScript library available on the web[12].

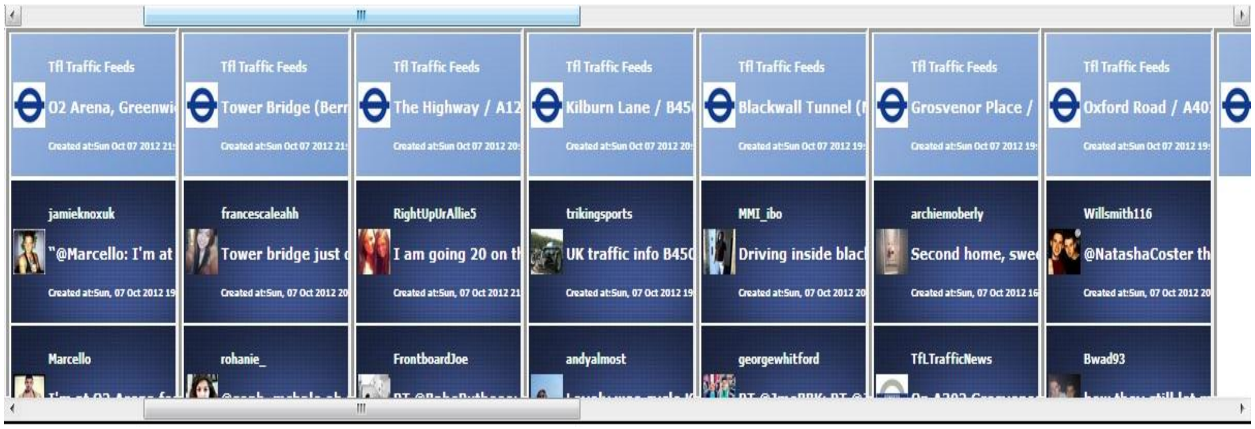
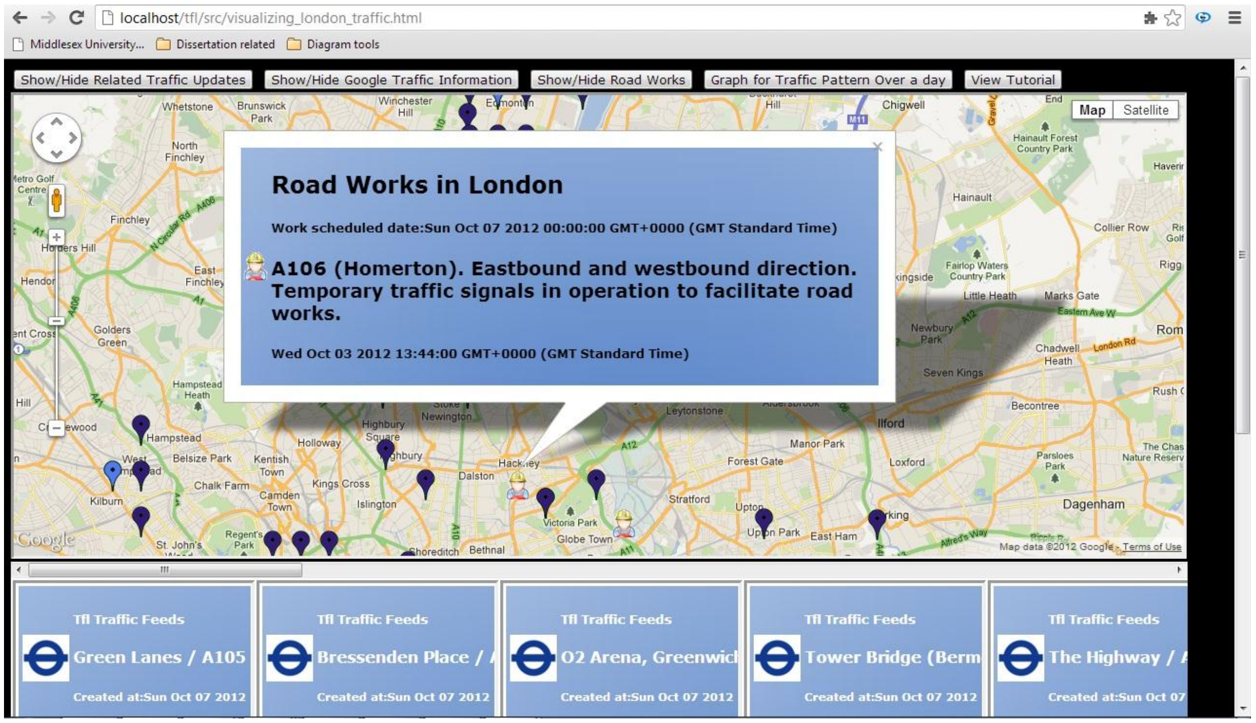


Figure 9: Dynamically create DIV

[12] http://rocha.la/jQuery-slimScroll

6.1.5. VIEWING ROAD WORKS IN LONDON

The Road works in London is a vital information when comes to the prediction of the traffic condition. So they are given prior importance and are categorized separately from the normal traffic conditions. These are pre-scheduled traffic information made available by the Transport for London Data Service. Any information related to real time road work updates is displayed in the dynamically created div from the information panel of the user interface. When the user clicks on the road works button from the control panel the road works can be made visible or invisible. A pop-up window will be displayed when a click is made on the road works marker as shown in figure 10. It explains the place, description, scheduled time of work and last updated time for the road works. Also similar to the animation added to normal Google markers displaying the traffic updates, the road works is also animated to attract the users of the system.


localhost/tfl/src/visualizing_london_traffic.html
Middlesex University...
Dissertation related
Diagram tools
Show/Hide Related Traffic Updates
Show/Hide Google Traffic Information
Show/Hide Road Works
Graph for Traffic Pattern Over a day
View Tutorial
Map
Satellite

Road Works in London

Work scheduled date:Sun Oct 07 2012 00:00:00 GMT+0000 (GMT Standard Time)

A106 (Homerton). Eastbound and westbound direction. Temporary traffic signals in operation to facilitate road works.

Wed Oct 03 2012 13:44:00 GMT+0000 (GMT Standard Time)

Tfl Traffic Feeds
Green Lanes / A105
Created at:Sun Oct 07 2012

Tfl Traffic Feeds
Bressenden Place /
Created at:Sun Oct 07 2012

Tfl Traffic Feeds
O2 Arena, Greenwich
Created at:Sun Oct 07 2012

Tfl Traffic Feeds
Tower Bridge (Berm
Created at:Sun Oct 07 2012

Tfl Traffic Feeds
The Highway /
Created at:Sun Oct 07

Figure 10: Road works in London

6.1.6. HIDING/DISPLAYING USER TWEETS

As well explained in the implementation section, the User Tweets in the system will be placed in successive square patterns around the main traffic update markers. Sometimes these user tweets will be annoying if there is dense traffic information displayed on the map. So in order to solve this problem hiding/displaying the user tweets option is available as a button to be clicked, added to the control panel of the user interface. A better view of the marker pattern of the user tweets marker is perceivable once the user zooms into the Google map. To be noted there will be eight markers available in one square and if it exceeds the outer square accommodates another eight markers. By doing so many markers can be displayed by effectively making use of the Google maps space.

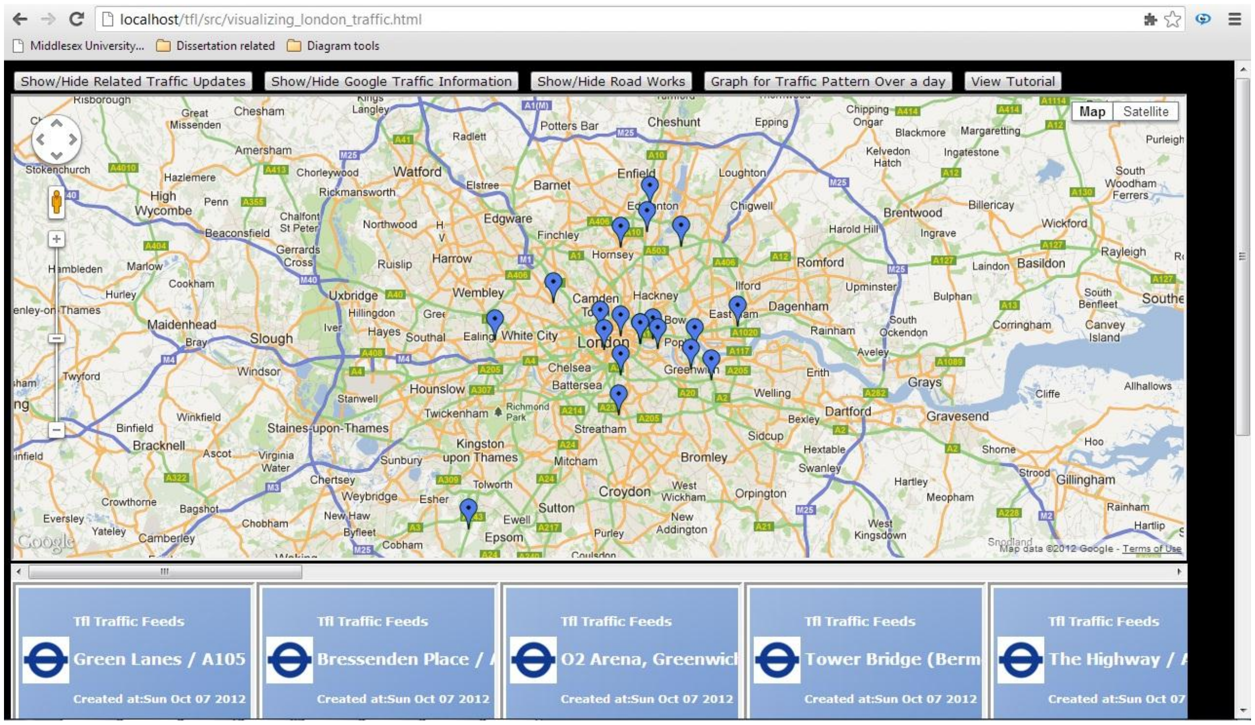


Figure 11: Hiding of user tweets

6.1.7. DISPLAYING GOOGLE TRAFFIC INFORMATION

The Google Traffic Information is well explained in the literature review section of the thesis. In order to make use of the Google Traffic Information click on the Show/Hide Google Traffic information button from the control panel of the User interface. The speed of moving traffic is depicted by color [13] in the Google Maps as shown by figure 12. The Google Traffic Information color description is as follows:

[13] http://support.google.com/maps/bin/answer.py?hl=en&answer=61455&topic=1687356&ctx=topic

- Green color: Indicates the speed of traffic more than 50 miles/hr. or 80 km/hr.
- Yellow color: Indicates the speed of traffic between 25-50 miles/hr. or between 40-80 km/hr.
- Red color: Indicates the speed of traffic less than 25 miles/hr. or 40 km/hr.
- Red/Black: Indicates very slow moving or dense traffic
- Gray: Indicates no data available at the moment.

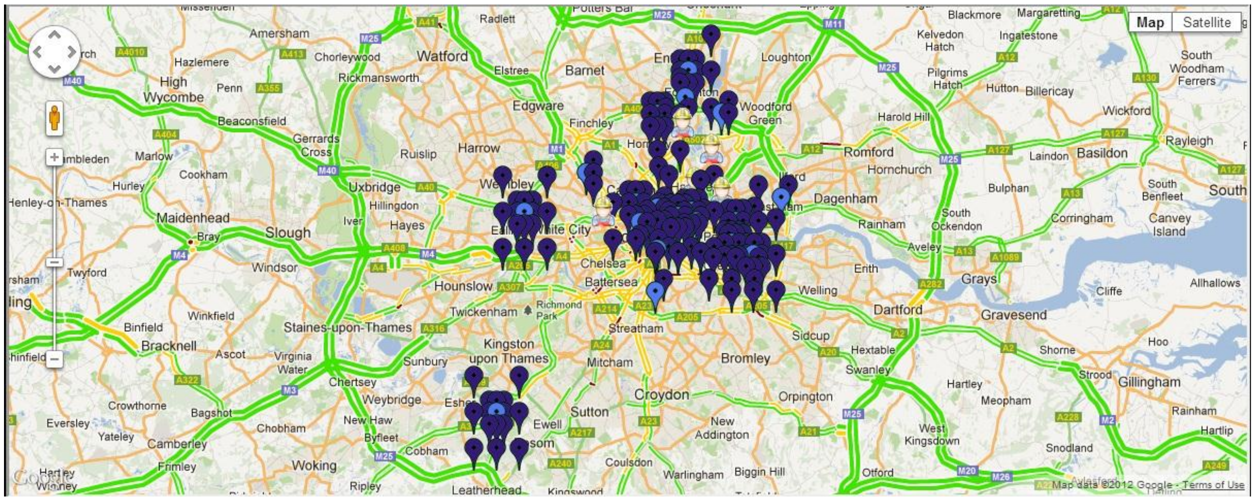


Figure 12: Google traffic information

6.1.8. UNDERSTANDING THE GRAPH

As explained in the earlier sections, the Graph is used to visualize the number of the traffic updates over a day in London. The Bar Chart is utilized from the pre-defined JavaScript Infovis Toolkit library and it incorporates readily available animations. To well understand the traffic updates, x-axis is split into six intervals 4 hours each covering the whole 24 hours for a day. The y-axis represents the number of main traffic updates and thus the traffic pattern over a day is made available to the user as shown in figure 13. The Graph panel is located below the information panel in non-viewable area of the screen. So in order to make the user to locate the Graph, a button for viewing the Graph is added to the control panel. When the user clicks the button the web page is scrolled down by JQuery animation thereby revealing the Bar Chart to the user.

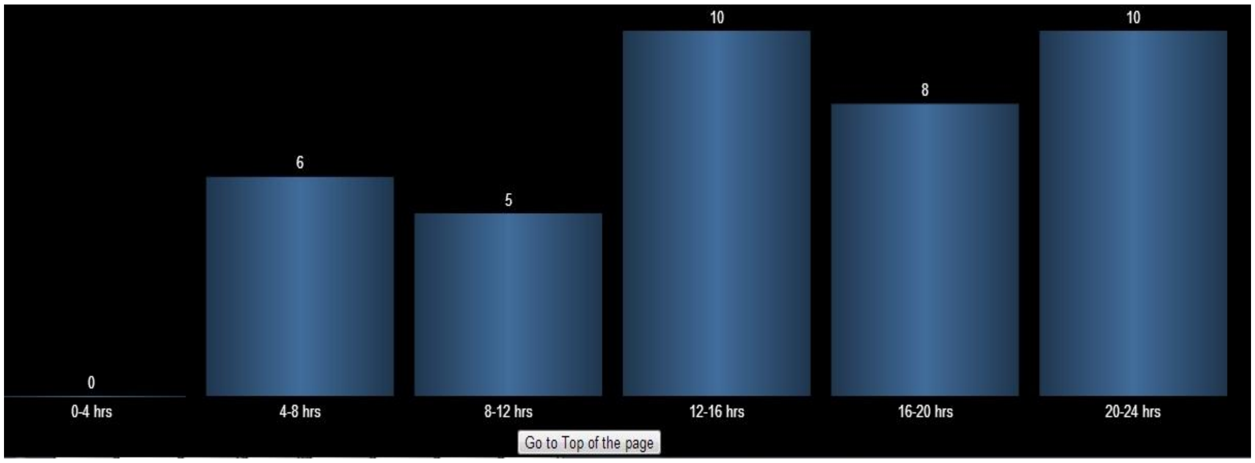
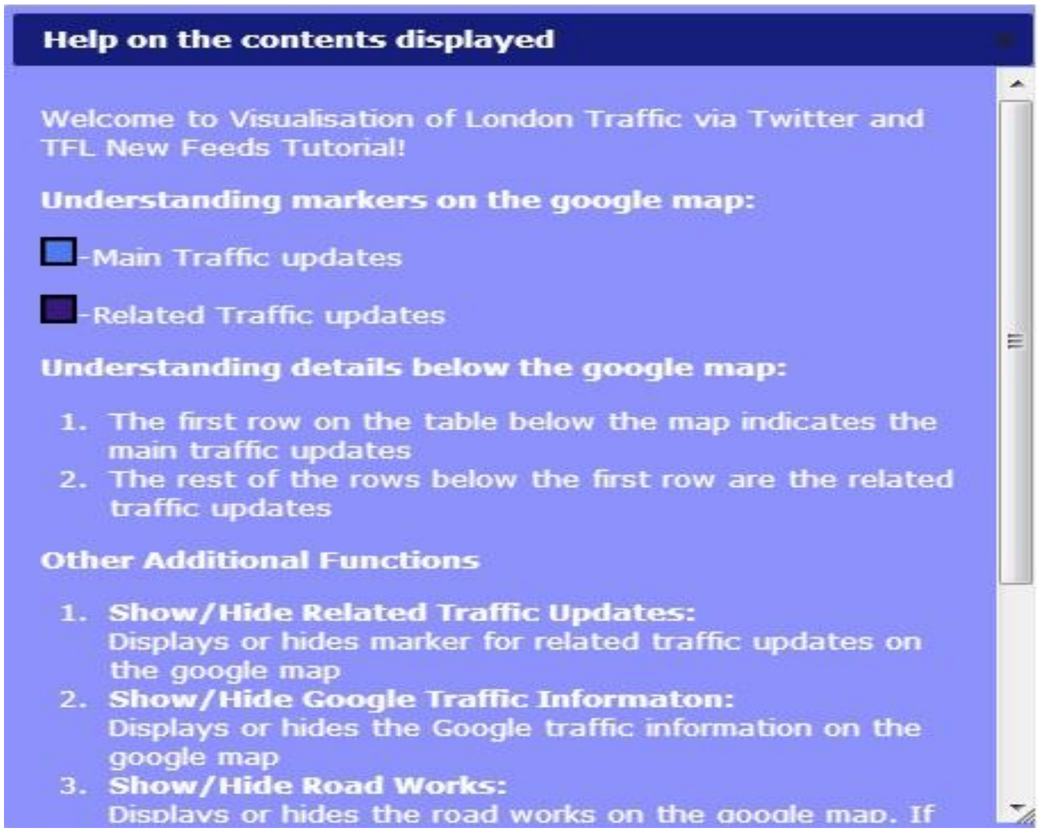


Figure 13: Understanding the Graph

6.1.9. GETTING HELP

After going through the demonstration section of the thesis if you still find it difficult to understand working on the system it is worth to have a look at the tutorial. The control panel has a 'view tutorial' button and when clicked, a modal dialog box designed using JQuery is shown. It briefs you the functionality of the system and will be short; not vague when compared to this demonstration section.



Help on the contents displayed

Welcome to Visualisation of London Traffic via Twitter and TFL New Feeds Tutorial!

Understanding markers on the google map:

🟦-Main Traffic updates

🟪-Related Traffic updates

Understanding details below the google map:

1. The first row on the table below the map indicates the main traffic updates
2. The rest of the rows below the first row are the related traffic updates

Other Additional Functions

1. **Show/Hide Related Traffic Updates:** Displays or hides marker for related traffic updates on the google map
2. **Show/Hide Google Traffic Informaton:** Displays or hides the Google traffic information on the google map
3. **Show/Hide Road Works:** Displays or hides the road works on the google map. If

Figure 14: Tutorial for user

6.2. EVALUATION

Testing the system alone won't ensure that the system is well developed and capable of fulfilling the requirements of the user. Evaluation is an important criterion to assess the performance and satisfaction of the user. By doing it helps in tracking the problems in the system and that are to be rectified in the future revisions of the end product. Further evaluation ensures that the scope of the system is well accomplished.

6.2.1. QUESTIONNAIRE

In order to carry out the evaluation process for the system, a set of questionnaire is prepared to assess the user's interest and satisfaction on the developed system. After the questionnaire is framed it is given to a group of 20 persons. To maintain the ethical aspect each user's feedback is made confidential.

The structure of the following questions framed is such that the first five questions evaluate the idea of user about the concepts used in the system. The remaining five questions determine the user's feedback on the working system. Based on their evaluation made, the following are the questions framed and the corresponding user's evaluation on the whole.

1. Have you heard about Online Social Networks?

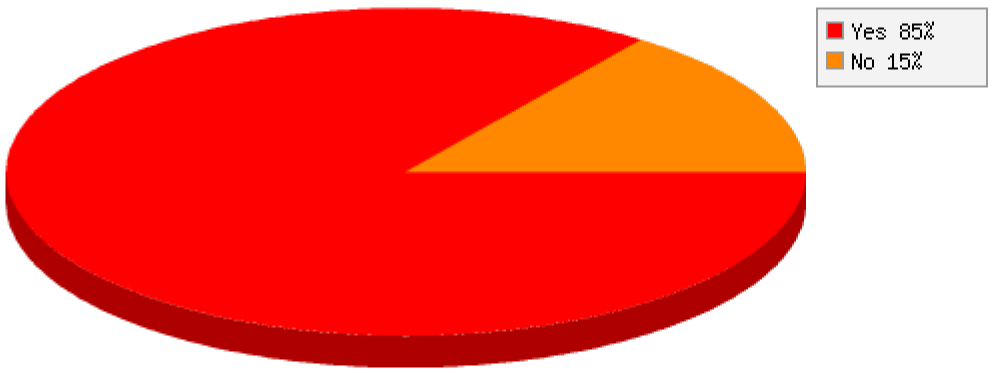


Figure 15: Question 1

2. Which Social Network do you prefer the most?

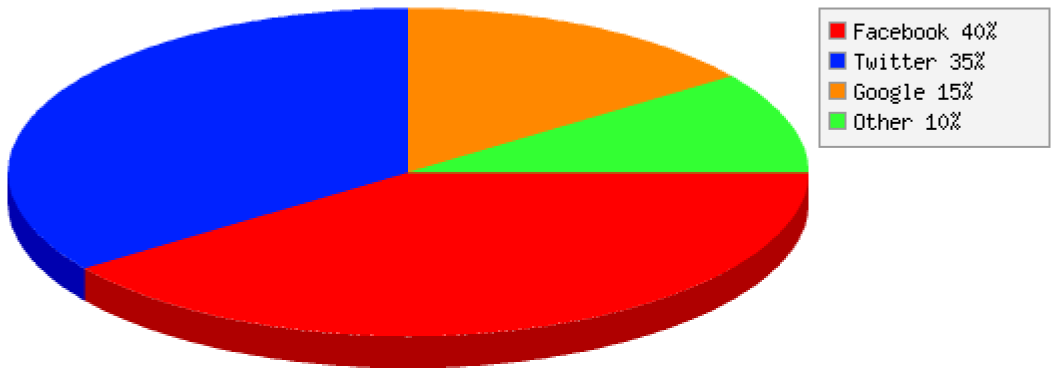


Figure 16: Question 2

3. Had you ever used Google Maps before?

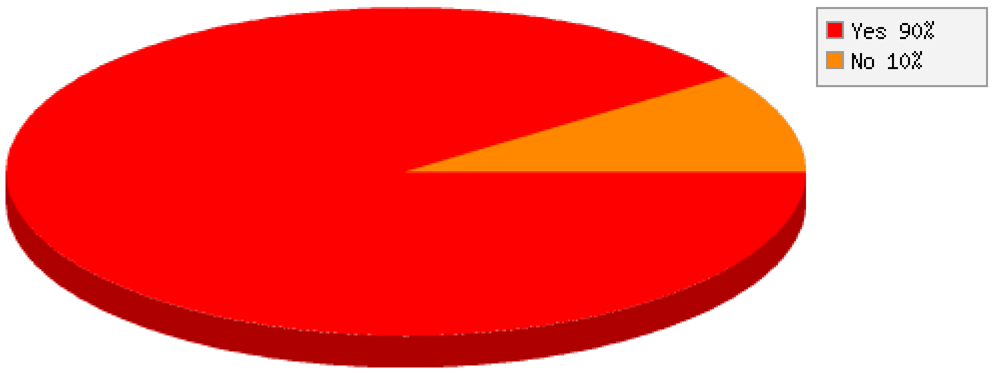


Figure 17: Question 3

4. Had you made use of any online resource for predicting traffic?

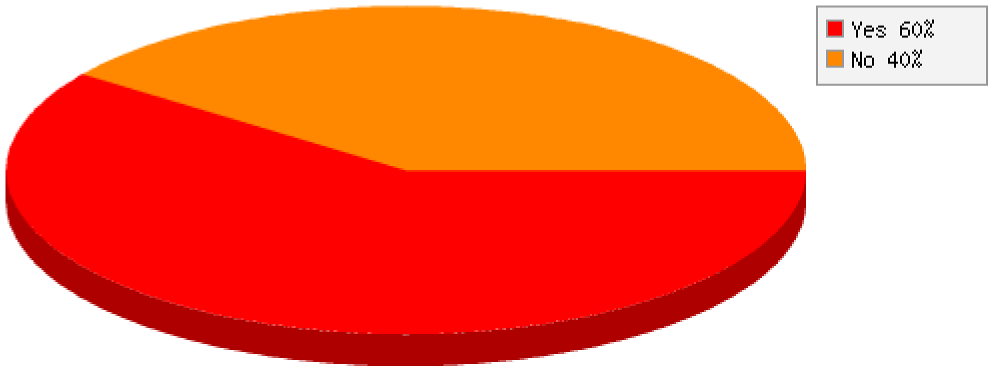


Figure 18: Question 4

5. Have you heard about the TFL traffic updates made by twitter?

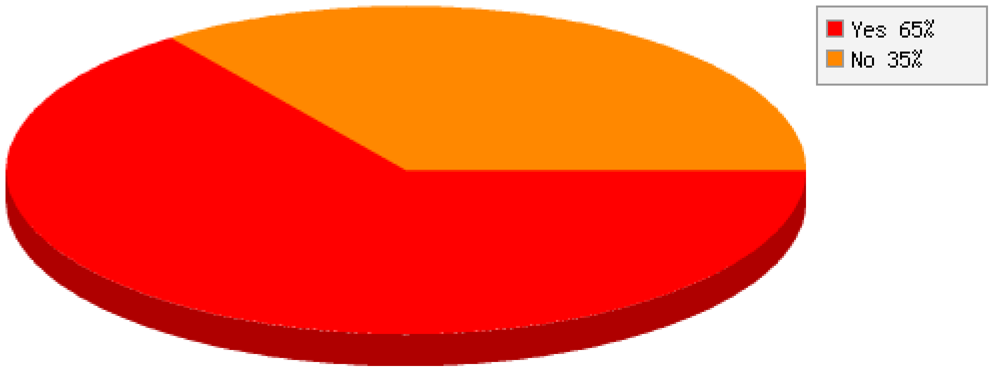


Figure 19: Question 5

6. How is the user interface design made?

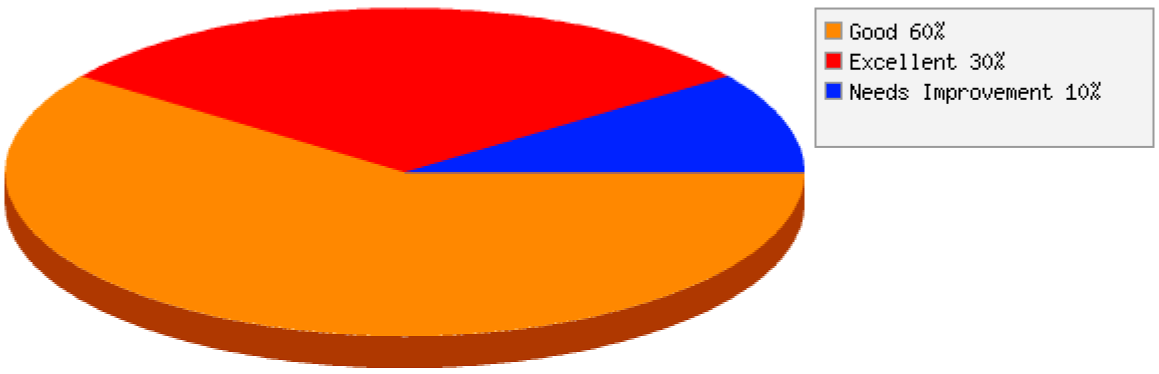


Figure 20: Question 6

7. How quick is the loading time of the system?

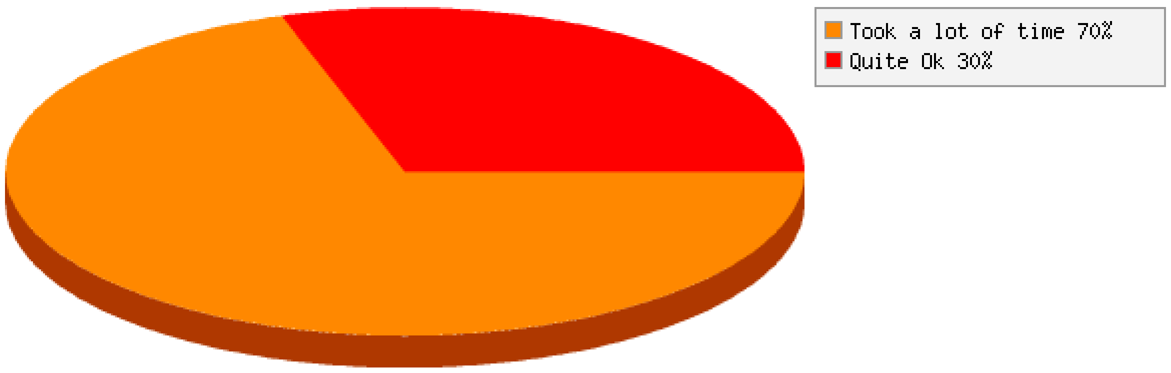


Figure 21: Question 7

8. Had you required any support while working on the system?

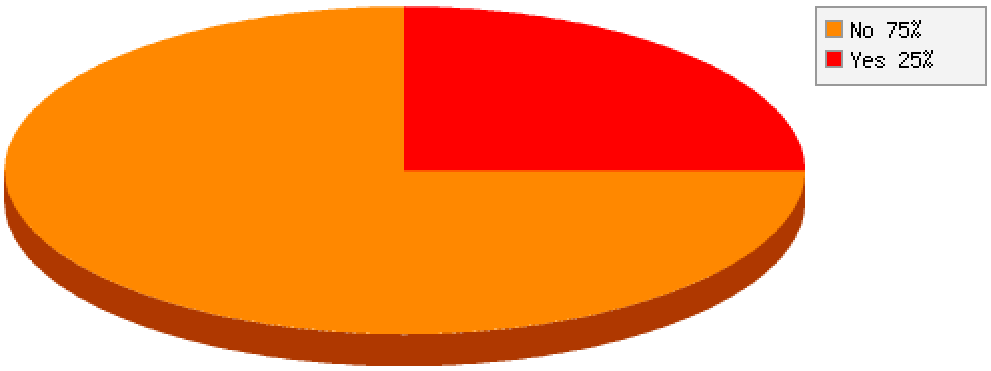


Figure 22: Question 8

9. Had you able to predict the traffic for your trip by using the current system?

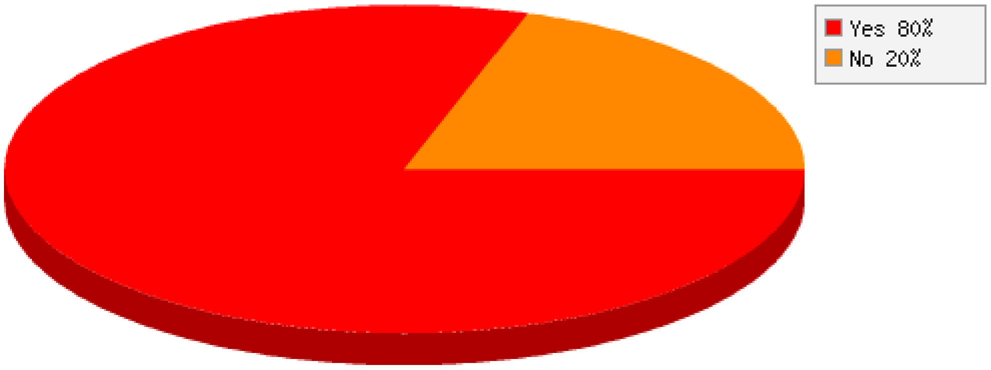


Figure 23: Question 9

10. Had the intended use of the system fulfilled?

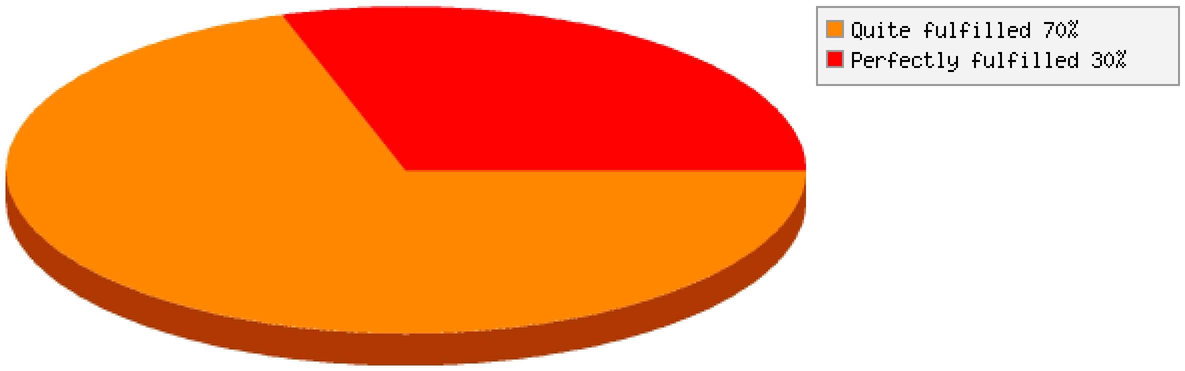


Figure 24: Question 10

7. CRITICAL EVALUATION

7.1. CRITIQUE OF MY WORK

The user evaluation and their feedback about the proposed system helped the author to further identify the areas in which his approach was lacking and the areas in which the design can be improved. Since the system was to be developed for the user and not for the developer, the author felt more consideration should be given to the users. This is a better way of evaluating the performance of the system. Based on the evaluation made by the users on the system and on the author's evaluation of the system, the following are the list of critical evaluations made on the newly proposed system.

7.1.1. EMPHASIS ON LITERATURE REVIEW

Initially at the time of generating the idea for the proposed system, the author found tough to cope with complex design concepts. So gradually he started concentrating more on the proposed system. Eventually when the author progressed towards the attainment of the scope, the concepts became familiar to him. Then the author started to study more about the already existing system and this helped a lot during the progression of his project. At the end the author realized if he had given more importance to the literature review right from the beginning, more enhancements could have been made on the proposed system.

7.1.2. PROPER UPDATE ON CURRENT TECHNOLOGY

It is vital to keep track on the current technological advancement made on the area of research while trying to develop a system. After a brief literature review the author's focus was more on the implementation of the project rather than having a look at the existing system designed on its scope. Later the author started realizing that there was an existing system designed by TFL that shows live traffic visualization for London. So he started altering his system design to implement concepts that made the current system standout from the visualization by TFL. Eventually he accomplished it, but the design could have been even better if the author would have kept a track on the current technology in the area of research.

7.1.3. SELECTING THE PROGRAMMING LANGUAGE

The choice of the programming language is vital for the development of a system. A detailed research should be carried out on the selection of the programming language before proceeding towards implementing the system. Initially while the author was setting up the basic layout of the system everything seemed working fine. But later when he progressed towards implementing complex features for the system he began to feel the heat. Especially for extracting location from the retrieved TFL traffic news updates and TFL Online syndicated feeds je found himself in need of a library that does 'Named Entity Recognition (NER)'. But when the author started searching on the web he couldn't come across a library in JavaScript that could implement the NER better. Then he manually started retrieving the roads from the traffic updates. So this situation could have eliminated if the author selected a programming language that had a good support and that best suites his system requirements.

7.1.4. TIME MANAGEMENT

Time Management is vital in almost all areas of the day-to-day life for every human. So it is better to plan ahead in advance rather than realizing at the end and feeling distressed. Even the same applies in the area of software development. Extra caution should be taken in the information technology sector for managing time in order to compete with other companies in the market. In the current project development the author experienced a slight problem with time management. However he was able to attain the desired scope of the system design, the system could have been well-polished if he would have divided his work accompanied with a better time schedule. So prior to the project development it is important to have an application like Gantt chart to better generate a proper time schedule on the task to be accomplished.

7.1.5. MORE FOCUS ON OBJECT ORIENTED DESIGN

The object oriented design helps in creating a simple, extendable, maintainable and re-usable system. But in our current design less focus was given to develop the system in an object oriented design. Even though Google maps incorporates the object oriented design, the rest of the system was not focused on OOPS design. In addition the author encountered a problem in generating the UML diagram due to non-compliance of the system to a proper object oriented design.

7.1.6. INSIGHT ON DATA AND RATE LIMITATION

Data limit and rate limit are two important criterions when it comes to the development of a system. It affects the quality and performance of the end product if these criterions are not taken into account. In the author's system design, the twitter API had a data limit of 200 tweets per hour and the Google geo-coding API had a specific rate limit. Since the TFL traffic news updates won't exceed more than 200 tweets per day the data limit was not a problem for the author's proposed system. But for the Google geo-coding API the author selected a best replacement by using the Yahoo Place Finder API. If these factors were not taken into account, his system design would have been a failure because it relies heavily on data density and execution speed. Since the data and rate limits were properly analyzed before beginning of the system design, future problems were avoided to a great extent.

7.1.7. IMPROVING PROGRAM LOADING TIME

The execution time for a system is also a critical factor and will drastically affect the performance of the system. In the author's design the main cause for the delay in the loading time of the program was due to the asynchronous execution of the JQuery AJAX loop. To be noted the asynchronous execution of the JQuery was encountered while requesting the geo-coding API. Due to the asynchronous execution of the JQuery AJAX loop, the sequential execution of the JavaScript program was hindered. So to solve the above JavaScript timers was used and hence the overall loading time of the system was increased. A better programming solution could have been used to solve this loading time problem.

7.1.8. SELECTION OF GEO-CODING API

The choice of the geo-coding API is important when it comes to system that relies heavily on geographical locations. Since the author's proposed system was based on depicting traffic condition in London, tracing the location of the traffic impact remained more vital. This in turn depends heavily on the geo-coding API. Although the author had managed to implement his system for showing traffic condition for London, still he couldn't refine the position of the Google marker to the exact incident location. However the author had used the best available geo-coders namely the Google and Yahoo geo-coders, still he couldn't find an exact solution to refine the position of the incident location.

7.1.9. FILTERING USER TWEETS

In the author’s proposed system after extracting the roads from the main traffic updates, he fed those roads to twitter SEARCH API to find the related tweets on the roads. In the returned search results there were noisy information tweeted by different twitter users that are not related to the traffic condition. The author was not able to find a suitable algorithm so that the noisy information could be filtered and removed from the system. If this was problem was solved by a suitable solution it would have improved the quality of the system to a greater extent.

7.2. KNOWLEDGE ACQUIRED

To the best of the author’s knowledge that he had earlier in the area of software development, he improved a lot by doing this Master’s dissertation. Before beginning the project he was quite familiar in working with JavaScript and HTML. But by doing this dissertation he learned how to add JavaScript libraries to his web application. Eventually by doing so he came across JQuery JS library for the first time. After the completion of his coding part the author felt that he had learnt a lot in the practical application of JQuery and even with JQuery UI library. In addition the procedures for the configuring the web server was also covered at the time of implementation.

Beside the programming language, the author was able to learn in detail about the different open-sourced Twitter Application Interface (API) and the way in which they can be accessed via JQuery AJAX request. Also not only the twitter API, he was able to learn about Google Maps API, Yahoo Place Finder API etc. In addition by this dissertation the author came to know about the TFL open-sourced Online Syndicated Data Feeds and the way to access it. The concept of JavaScript Object Notation (JSON) array and the technique of parsing the XML file were also covered by him in the meantime. Overall it was a great learning experience combined with an excellent guidance from his supervisor who gave a lot of suggestions and idea to best design this newly proposed system.

8. CONCLUSION

8.1. THESIS CONCLUSION

Twitter always remains a valuable data enriched source to be data mined to serve wide variety of purposes. By the newly proposed system the raw traffic updates retrieved from twitter was successfully categorized based on the roads and made available for the users; so that they can easily predict the traffic conditions specific to the roads in London. Rather than spending more money in maintaining a traffic sensor system, the technique of using existing data to predict road traffic is cheaper and remains a more effective solution. Further by providing detailed information on the traffic to the users, they can best utilize them to decide their traffic free route. But however more research has to be made on the areas of text analysis for extracting geographic location and filtering noisy information. To further increase the importance of the newly developed system's approach, the transport for London has linked almost all transport related facilities to twitter.

8.2. FUTURE WORK

The transport for London is increasing its use of twitter in almost all areas of transport and even in all modes of travel. Recently they had added the facility to answer the oyster related queries for London via twitter. In the area of real time information the twitter accounts are to be created to cover trams, buses and even the Airlines of Emirates. Other related real time transport updates for London include information for helping the journey through a range of services covering the variety of operating on the rails.

Since more and more importance is given to twitter, the currently developed system to predict traffic in London can be further extended to provide real time updates for buses, trams, tubes etc. Also even more reliable sources can added to the system to improve the data density and accuracy of the system.

REFERENCES

[1] Endarnoto, SK, Pradipta, S, Nugroho, AS & Purnama, J 2011, 'Traffic Condition Information Extraction & Visualization from Social Media Twitter for Android Mobile Application', *Proceedings of the 2011 IEEE Conference on EEI (Electrical Engineering and Informatics),* pp. 1-4.

[2] Syndication Developer Guidelines: Transport for London Data Service 2012, *Being Prepared for Syndication Developers*, Transport for London, Retrieved September from http://www.tfl.gov.uk/developers

[3] An Information Network 2012, About Twitter. Retrieved September 2012 from http://twitter.com/about

[4] Kobie, N, Google gets live traffic information from Highways Agency 2008. Retrieved September 2012 from http://www.itpro.co.uk/607479/google-gets-live-traffic-information-from-highways-agency

[5] Belmonte, NG 2010, JavaScript InfoVis Toolkit. 17 September 2010. Belmonte, NG*: Blog*. Available from: <http://blog.thejit.org/2010/09/17/javascript-infovis-toolkit-webgl >. [30 September 2012].

[6] Wu, YJ, Qian, D & Wang, Y 2007, '**A Google Map Based Arterial Traffic Information System**', *Proceedings of the 2007 IEEE Conference on Intelligent Transportation Systems, Seattle, WA, USA*. pp. 968-973.

[7] Tiemann, M, Open Source Initiative 2006, History of the OSI. Retrieved September 2012 from http://opensource.org/history

[8] The Solution for Maps Application for the Desktop and Mobile 2012, Google Maps JavaScript API v3. Retrieved October 2012 from https://developers.google.com/maps/documentation/javascript

[9] Bell, D 2003, ‘UML basics: An introduction to the Unified Modeling Language’. Retrieved October 2012 from http://www.nyu.edu/classes/jcf/g22.2440-001_sp06/handouts/UMLBasics.pdf

[10] Siddique, Q 2010, ‘Unified Modeling Language to Object Oriented Software Development’, *International Journal on Innovation, Management and Tech.*, Vol. 1, No. 3, pp. 264-268.

[11]Dork, M, Gruen, D, Williamson , C & Carpendale, S 2010, '**A Visual Back Channel for Large-Scale Events’**, *IEEE Transactions on VCG(Visualization and Computer Graphics),* Vol.16, No. 6, pp. 1129-1138.

[12] Yin, J, Lampert, A, Cameron, M, Robinson, B & Power, R 2011, ‘Using Social Media to Enhance Emergency Situation Awareness’, *Proceedings of the IEEE conference on IS(Intelligent Systems),* pp. 1-7.

[13] Murthy, D, Gross, A & Longwell, S 2011, ‘Twitter and e-Health: A case study of visualizing cancer networks on Twitter’, *Proceedings of IEEE conference on Information Society*, pp. 110-113.

APPENDICES

WAMP Server

A WAMP Server is a software package that exists independently and includes various sub-components. WAMP is an acronym and is derived from Windows-Apache-MySQL-PHP, Perl or Python. It is released for the purpose of configuring the Web Server with an ease.

Procedures for installing and configuring WAMP Server

- Visit the WAMP Server website and download the application based on the OS used.
- Once the .exe file is downloaded, the appropriate file is double clicked and given instructions is to be followed.
- Without any manual configurations the installation process executes automatically based on the opted user prompt.
- The different sub-component version depends on the WAMP Server versions.
- Once the installation gets completed navigate to the www folder in which the WAMP Server is installed and copy the PHP or HTML to be accessed via local web server.
- Then navigate via the start menu if you are using Windows and click on the start WAMP server.
- Then open the browser and type the URL http://localhost/ or http://127.0.0.1/ to test your local web server.
- If successfully configured you will get the WAMP Server configuration page or else try getting online solutions.

JQuery UI

JQuery User Interface is a set of features that interacts with user and is built with a proper base by the jquery, which is the JavaScript library. It is written with a motive of enhancing the interface to have a better appearance and feel. The recently released version available on the web is made use of the current web application. The modal dialog, progress bar used in the current system are designed via the Jquery UI.

JQuery AJAX

The JQuery incorporates a full library support for the Asynchronous JavaScript and XML (AJAX). The JQuery has a numerous method to enhance AJAX requests and even enable the AJAX requests without refreshing the web browser.

JSON

JSON is the JavaScript Object Notation and is light weight format that can be utilized to interchange the data. It is easy for reading and writing these JSON arrays by humans and even easier for machines to process it. The other striking benefit for the JSON arrays is that it is programming language independent. Due to its various significance most of the open-source servers or API are designed to interchange their data via JSON formats.