7CCS4PRJ Final Year Individual Project

Real-Time Visualisation of Bus Delays in London using AVL Data

iBus Disruption Monitor

A project in collaboration with Transport for London

Final Project Report

Author: Konstantin Vladimirov Draganov

Supervisor: Dr Steffen Zschaler

Course: MSci Computer Science

Student ID: 1101314

September 2014 - April 2015

Abstract

Automatic Vehicle Location (AVL) systems for bus fleets have been deployed successfully in many cities. They have enabled improved bus fleet management and operation as well as wide range of information for the travelling public. However there is still a lot of area of utilisation of the data that these AVL systems generate and produce. This report explores and analyses the tools and applications currently available at Transport for London (TFL) bus operation unit. The report then proposes a prototypical tool for monitoring the bus delays in real time in the network. This tool offers objective source of processed information to bus operators and control room staff. However further work need to be done in order to place this tool in production environment as urban delay detection is very complex and unpredictable as will the report justify.

Originality Avowal

I verify that I am the sole author of this report, except where explicitly stated to the contrary. I grant the right to King's College London to make paper and electronic copies of the submitted work for purposes of marking, plagiarism detection and archival, and to upload a copy of the work to Turnitin or another trusted plagiarism detection service. I confirm this report does not exceed 25,000 words.

Konstantin Vladimirov Draganov

September 2014 - April 2015

Acknowledgements

First and foremost I offer my sincerest gratitude to my supervisor Dr Steffen Zschaler for the continuous support, guidance, encouragement, insightful comments and hard questions throughout the course of this exciting project.

I would also like to thank Andrew Highfield and Keith Elliot from TFL for providing me with all the needed data, information and feedback which has been immensely helpful.

Last but not the least, I would like to thank my parents Vladimir and Veselka also my sister Lilia and my partner Simona for their support and patience not only during the project, but throughout my life as well.

Contents

Chapter 1

Introduction

1.1 Motivation

London bus network is one of the largest and most advanced bus networks in the worlds. It is responsible for more than 2.4 billion passenger journeys a year [8]. The constant population growth of England's capital has been also driving the expansion and improvement of the transport networks across the city. Transport for London (TFL) is in charge of its operation and its bus network if recognised as one of the top in the world in terms or reliability, affordability and cost-effectiveness [8]. The capital's bus network is continually increasing along with the city's population [11]. Maintaining such a large scale network requires careful planing and monitoring. Being able to maintain such high reliability service 24/7 364 days in the year requires employing new technologies. This also helps keep costs down and thus keeping the service more affordable and accessible for the general travelling public. Each bus in the TFL network has been equipped with state of the art GPS enabled automatic vehicle location (AVL) system named iBus[3]. This AVL system has led to improved fleet management and has enabled the creation and improvement of multiple applications [13]. The system is generating large sets of data both in real-time as well as historical data. This helps, the bus operators and the emergency control room at TFL responsible for maintaining the bus network

(CentreComm), to better manage and maintain the smooth operation of the bus network.However there are currently problems for which CentreComm staff are required to manually analyse these data sets in order to discover in which parts of the networks problems are occurring. This is because there is lack of readily available preprocessed information. This is very impractical and time consuming and currently they ultimately rely on individual bus operators and drivers to notify them of possible problems. Once alerted of a possible disruption in the network they can start their own investigation into first verifying what they have been told by the bus drivers/operators and then into finding the cause and the actual severity of the problem. This often could lead to spending time and resources into investigating non existent problems. This is where this project comes in place to address this inefficiency and to propose, implement and evaluate a prototypical tool for real time monitoring of the bus network.

1.2 Scope

The scope of this project is to analyse the current work flow of CentreComm operators and their needs. The main goal is to design and implement a prototype which to aid the control room staff. This tool has to work and analyse the data that has been made available in real time. The main two problems that are in the scope of this project and need to be solved are:

- Analysing and identifying the disruptions in the bus network in real-time using the provided data.
- Producing a prioritised list of the disruptions classified using rules and heuristics gathered during meetings and discussions with the key stakeholders from TFL.
- Visualising the calculated list in an appropriate format.
- Evaluating the performance of the tool. This is essential as it need to be proven that the results could be trusted.

1.3 Aims

The main aim of this project is to design and implement a real-time visualisation tool which to highlight disrupted routes or parts of the TFL bus network where disruption might happen or have already happened even before the bus drivers or operators have noticed and alerted CentreComm. This could could be subdivided into two smaller aims:

- The first one which is independent of the other is to enable the processing of the data generated by the buses in the TFL's bus network. The tool need to be able to analyse the input data sets and calculate and output a list of the disruption that are observed in the network. It has to present information regarding the location (section) in the transport network and their severity.
- The second part of the main aim above is to visualise the generated output in an easy to use and understand way.

1.4 Objectives

The objectives that have been followed in order to successfully meet the above stated aims are:

- Getting in depth understanding of the problem and current work-flows that are in place.
- Researching similar work that has already been done and how it can relate to our problem.
- Obtaining samples of the available data and understanding what it means
- Gather and refine the user requirements during discussions and meetings with CentreComm staff and stakeholders.
- Design and develop initial prototype which to be further refined and improved after obtaining feedback from TFL.

- Test and evaluate that the tool works according to the user requirements and the design specifications.

1.5 Report Structure

In order to help the reader I have outlined the project structure here. The report would continue in the next chapter by providing the reader with all the background knowledge needed for the rest of the report. This would include brief of background on the current work-flow CentreComm operators follow and its inefficiencies. I will also give background on the iBus system and the data that the tool would need to operate with. I then explore related work that has already been done and how ours differs. This is followed by alternative approaches and models that could be utilised. Afterwards the report focusses on the specific requirements that have been identified and gathered from CentreComm. The report then goes on to discuss the design and the implementation of the proposed system. This is followed then by Chapter 5 and 6 which address testing and evaluation of the prototype. I conclude the report with a summary of what has been achieved and guidance how the work presented in this report could be further developed and improved.

Chapter 2

Background research

This chapter aims to introduce some concepts surrounding our problem domain, which to help the reader to understand and follow easier the following more technical chapters. I also present a review of the related work that has been done in this area. The below sections look at some of the key aspects and problems that arise. I then conclude by providing a number of alternative methods for solving our problem.

2.1 London Bus Network

London bus network is one of the most advanced and renowned in the world. It runs 24 hours and it is extensive and frequent. Every route in the network is tendered to different bus company operator [5]. These operators agree with TFL that the routes they are operating would be served either according to a predefined fixed schedule (e.g. a bus stop need to be served at 1pm, 3pm etc.)or on a headway (e.g. a bus stop need to be served every 5 minutes). However under different circumstances some delays occurring on a given route are beyond the control of the different bus operator companies. A simple example could be a burst water/gas pipe on a street used by a bus route or any other incident (even terrorist attacks [7]) and even simply a severe congestion. In situations like this bus operators have no authority or power to overcome such problems on their own. They can only ask CentreComm to intervene. CentreComm can do so by for example implementing a short/long term diversions or curtailments (short turning) some of the buses on the affected routes.

Buses in the network can be classified by multiple factors, however for the purpose of this report the main distinction we need to consider are high and low frequency bus routes. High frequency routes are routes where there are 5 or more buses per hour attending a given bus stop. Low frequency routes are those that have 4 or less buses alighting at a stop.

2.2 CentreComm

CentreComm is TFL's emergency command and control room responsible for all public buses in London. It is has been in operation for more than 30 years [7] and it employs a dedicated team of professionals who work 24 hours 364 days in the year. They are dealing with more than a 1000 calls on a daily basis. The majority of these calls come from bus drivers or bus company operators regarding problems and incidents happening within the bus network. CentreComm staff implement planned long and short term changes in the bus network in response to different events taking place in the capital (including the 2012 Olympics). They are also responsible for reacting in real time to any unexpected and unpredicted changes and disruptions, maintaining the smooth, reliable and sage operation of London busy bus network.

London bus network consists of around 680 bus routes operated by more than 8000 buses [1]. Each of this buses is equipped with state of the art iBus system to help monitor and manage this enormous fleet. CentreComm's way of operation has been transformed beyond recognition since it has first opened and today. It started more than 30 years ago [7] and it consisted of a couple of operators equipped with two way radios and pen and papers. Today CentreComm operators make use of numerous screens each displaying interactive maps (displaying each bus location) and CCTV cameras in real-time. However there is still a lot of room for automation and improvement in their way of operation in order to effectively and efficiently maintain the growing bus network.

2.3 iBus AVL

Automatic vehicle location systems make use of the Global Positioning System (GPS) to enable the remote (using the internet) tracking of the locations of the vehicles in a given fleet. This system combines a number of technologies including GPS, cellular communications and more with the goal of improving and cutting the cost of fleet management.

All of London buses operating on the TFL bus network have been equipped with state of the art and award wining [4] AVL system named iBus [6]. This system has opened a range of new applications that could be built on top of it using the information that is made available. The iBus system consist of a number of computer and communication systems, sensors and transmitters as described in [9] and [13]. One of the key components of the system on-board unit (OBU) which mounted on each of buses in the TFL bus fleet and consists of a computational unit connected to sensors and GPS transmitters (see figure 1 below taken from [9]). This OBU is responsible for a number of tasks including a regular (approximately every 30 seconds) transmission of the bus location. This information is currently used by the different bus operators for fleet management as well as by CentreComm for real-time monitoring of the buses and their locations. There are have been a number of other applications and systems that have been implemented and put in to use as a result of the data that is generated by the iBus AVL. Some examples include Countdown (real-time passenger information), improved bus priority at traffic signals and more [9]. This has led to improved and more affordable transport service.

CentreComm operators have access to an online system showing each bus location in the network on a map in real-time. This system allows them to see whether a given bus is behind, ahead or on time according to its schedule. However this does not show or alert the control room staff if a bus or a route

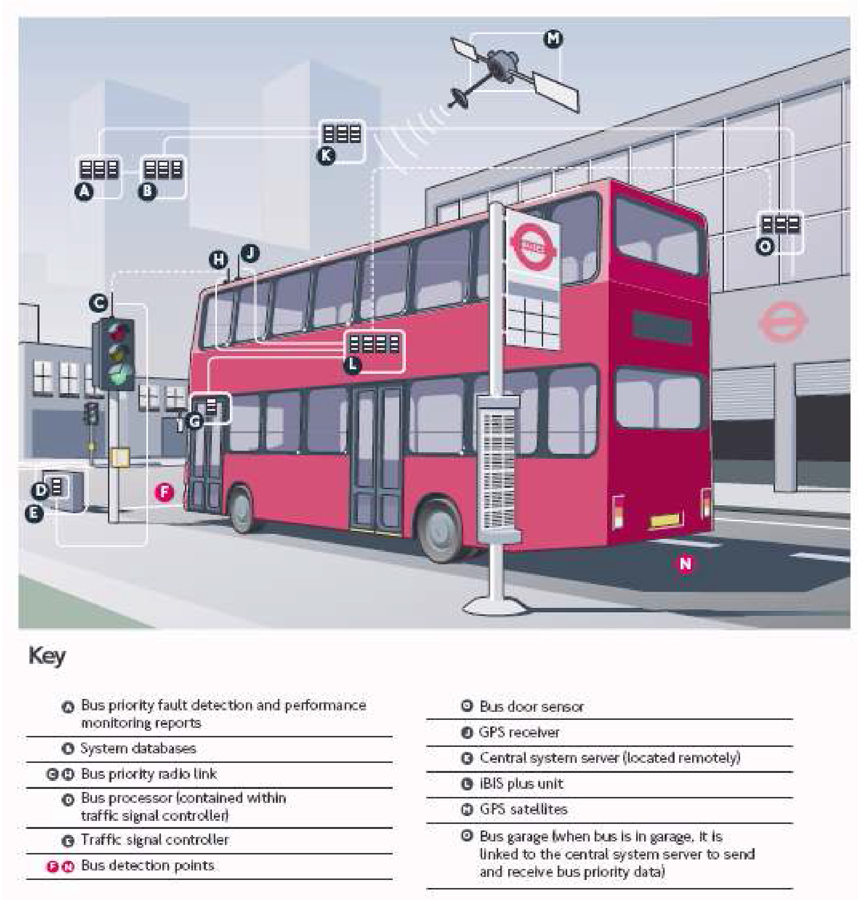

Figure 2.1: Overview of iBus System (Source: Transport for London, 2006a)

is disrupted. Control room staff also can see when was a given bus expected to arrive at a given bus stop and when it actually arrived, but this again is only per individual bus. Currently the work-flow is such that CentreComm need to go and analyse all this information manually (once bus drivers or bus company operator have contacted/alerted the control room) in order to figure out if a there is a problem and how severe it is. This is very inefficient, tedious and error prone process. Here is where our project comes into place to address the lack of preprocessed information.

2.4 Related Work

The literature is full of research towards accurately predicting bus arrival times [REFERENCE]. Also there is a lot of work done toward detecting calculating travel times in non urban environment. This include approached based on AVL probe data and automatic vehicle identification (AVI) as well as induction loops [REFERENCE]. Recently there is increased interest in detecting traffic congestion in urban areas using AVL equipped fleets as probes. However there are significant challenges due to the nature of the urban environment it self. Densely populate areas are influenced by many factors which could be affecting the general traffic flow. Another problem posed by urban environments is the irregularity of the AVL data transmission because of for example weak or lost signal at times (e.g. due to high buildings) or even there could be some noise which reduces the accuracy of the stated location.

However there is little research to my knowledge which address the issue of detecting disruptions and delays in a given vehicle fleet in urban environment. The literature is abundant of research on addressing the issues of bus prioritisation at traffic signals and junctions [REFERENCE SOME PAPER], but this is not directly related to our problem domain and thus is not discussed further in this report. Probably the most closely research that could be related to our problem is the one of detecting traffic congestion in a given network using AVL equipped buses as probes [Statistical Modelling and Analysis of Sparse Bus Probe Data in Urban Areas]. However from the literature [CITE You Are the Traffic Jam: An Examination of Congestion Measures] we can see that it could be difficult to precisely define what we mean by congestion in a transport network. It seems that congestion could mean various things to different studies and people.

For this reason in order to address our problem domain however we are not interested specifically in the general congestion as this could for example not affect the operation of the TFL bus fleet directly. A simple reason for this could be because there are significant amount of street which have a designated bus lanes which are prohibited for use by the general traffic. However there are studies showing that even under such circumstances there is a correlation between the general traffic flow and the performance of the buses in the same network [REFERENCE]. For our project we are interested however only in detecting the disrupted routes (sections of routes) and the severity of the disruption in the TFL bus network. We are interested in monitoring and detecting delays that are occurring in the network which would be classified, beyond the control of individual bus operator, by CentreComm.

However as other research has pointed out [REFERENCE] calculating travel time as measure for congestion is difficult task and it is very dependable on the environment and its conditions (e.g. weather, time of day, public demand etc.). For this reason and because of the data available this project would not try to measure disruptions by calculating travel time or bus speeds.TFL has provided us with example of the AVL data which among other things contains a the GPS coordinates of the bus at a given point in time and preprocessed deviation from the schedule value. For the rest of this chapter we assume this value is accurately calculated and that we would receive this value for each bus in the network at some regular interval. The provided data is discussed in further details in Chapter 5.

The literature review that has been undertaken as part of this project has showed that there are multiple approaches that could be undertaken to solve the problem posed by this project. There could be found different classifications of the models for predicting bus travel time[12]. According to [10] we could classify them into four computational models:

- Based on historical data
- Statistical
- Kalman Filtering
- Machine Learning

We could also add hybrid model which takes a combinational from the above [REFERENCE].

*****TODO:Include the table from lin.ppt

We could simplify our problem and explore different methods for performing time series analysis over the data. This would fall into the statistical category.

Time series is a sequence of data reading taken during successive time intervals. Time series analysis is performed on time series data in order to extract some meaningful statistics from the data. In addition to the time series analysis time series forecasting could also be performed to come up with a prediction for the next period in time based on what has been observed in the past. In our problem this would mean having a number of bus readings we could analyse them and come up with prediction of what would be state in the next point in time. There also different smoothing techniques which would remove random abnormal fluctuation (e.g. an incident with a single bus).In the below sections we look at a number of different approaches to analysing and forecasting the next state of the network from the available data in real-time (we update our forecast whenever more data becomes available).

2.5 Moving Averages

Moving average also called rolling or running average is a statistical calculation method. It helps to analyse data series by calculating a series of averages of subsets of the data. It can also be referred to as rolling or moving mean and it acts like a filter. Moving averages is commonly used in time series data analysis. It is used to smooth out any short-term fluctuations and derive some longer term trend or cycle.

2.5.1 Simple Moving Average

Simple moving average (SMA) is calculated by adding all the observations for a given period of time and dividing this sum by the number of observations. The simple average is only useful for estimating the next forecast when the data does not contain any trends. If data has some trend it is better to use another model which would take this into account. This technique weight's equally all data points. If we consider shorter term averages they would react quicker to changes while long term averages would have greater lag.

$$Include an example equation$$

2.5.2 Weighted Moving Average

The problem with the simple moving average is that it weighted all data points equally. Meaning that both older and newer data would have the same effect on the average. This however is not the case when using weighted moving average (WMA). In WMA model each data point would be weighted differently according to when the observation was made. For example if we consider a 5 period moving average we could multiply the oldest observation by 1 and the newest by 5. This would mean that recent data have bigger impact on the result.

$$Include an example equation$$

2.5.3 Exponential Moving Average

Exponential smoothing was first suggested by Robert Goodell Brown in 1956,[Brown, Robert G. (1956). Exponential Smoothing for Predicting Demand. Cambridge, Massachusetts: Arthur D. Little Inc. p. 15.] and then expanded by Charles C. Holt in 1957.[Holt, Charles C. (1957). "Forecasting Trends and Seasonal by Exponentially Weighted Averages". Office of Naval Research Memorandum 52. reprinted in Holt, Charles C. (JanuaryâĂŞMarch 2004). "Forecasting Trends and Seasonal by Exponentially Weighted Averages". International Journal of Forecasting 20 (1): 5âĂŞ10. doi:10.1016]

Exponential moving average (EMA) or also called exponential smoothing is very similar to WMA. The main difference is that in order to calculate it we do not need to keep all the data, but we could only store the latest value and forecast. Exponential moving average weights the data points exponential which means that the oldest data would have minimalistic effect on the result.There exist few exponential smoothing techniques including single, double and triple exponential moving average.

$$Include an example equation$$

2.5.4 Summary

If we compare the three types (simple, exponential and weighted) we can clearly see that the simple moving average offers the most smoothing however there is the trade-off of the biggest lag. The exponential moving average will react faster to changes and sits closer to the actual readings, however it might overreach at times. The weighted moving follows the movement even more closely than the exponential moving average. Determining which of these averages to use depends on your objective. If you want a trend indicator with better smoothing and only little reaction for shorter movements, the simple average is best. If you want a smoothing where you can still see the short period swings, then either the exponential or weighted moving average is the better choice.

2.6 Peak Detection

The detection of peaks and valleys in time series is a long-standing problem in many applications. Peaks and valleys represent the most interesting trends in time series. In the purpose to identify these trends in time series, we investigate two approaches of trend's detection. The first approach is based on a geometric definition of the trends. The second one uses a statistical definition of peaks and valleys. The two approaches are able to detect significant trends within time series. Nevertheless, only the statistical approach is able to find these trends in a global context. In this report we describe, define and illustrate algorithms of the geometric approach and the statistical approach. [TAKEN FROM - Survey of PeaksValleys identification in Time Series]

2.7 Other

Other approaches that have could be used include Kalman filtering [[1] Greg Welch, Gary Bishop, "An Introduction to the Kalman Filter", University of North Carolina at Chapel Hill Department of Computer Science, 2001] and Kalman Smoother, Markov Chains, Machine learning (neural networks), Bayesian networks. In this report we do not go into further details of these alternative as the aim of the project is to fully understand the requirements of the CentreComm and propose a prototype which to serve as a proof of concept.

2.8 Summary

For the purpose of this project and because of the data we have access to we have decided to implement the weighted moving average model for creating a prototype. The exact implementation details and decision that were taken are further discussed in chapter 5 along with detailed explanation on the available data.

Chapter 3

Requirements

In this subsection I have introduced and formalised the user and functional requirements. This is an important step of any project, especially computer science project,as it formalises the problems that the project is trying to address as outlined by the project aims and objectives in chapter 1. It also allows to be used as a measure for evaluating the success of the project once it has been completed. The requirements presented below have evolved and have been refined throughout the project lifetime in response to feedback and discussions carried out with the key stakeholders.

3.1 User Requirements

The user requirements provide a list of the functionalities that the user expects to be able to perform and see the according results, in the end product. These are what is expected from the system, but are not concerned how they are designed or implemented. The main user requirements are listed below:

- The tool must be able to produce a prioritised list of the disruption in the bus network that it has knowledge of.

- The tool must be prioritising the disruptions according to the user defined rules (these are still discussed and gathered from the user) The tool must be updating this list of disruptions whenever there is more data. This should happen as real-time as possible.

- The tool must be able to provide detailed information for every detected disruption. This has to include the specific section and route that are affected and its severity.

- The user must be able to interact with the system in order to lower or increase the priority of a given disruption (even ignore one).

3.2 Functional Requirements

These requirements specify in more details what the expected behaviour and functionality of the system is. They are built using the user requirements as an input and are detailed list of what the system should be able to accomplish technically.

- The system must have a representation of the whole bus network.

- The system must be able to read and process CSV files.

- The system must listen for new incoming data and process it in real-time.

- The system must be able to update it self whenever new data is detected.

- The system must be able to run without intervention 24/7.

- The system must keep track of the disruption detected and track how they evolve and develop.

- The tool must be able to keep information for a given window of time (e.g. data feeds from the last 2 hours).

- The tool must be able to output a prioritised list of disruptions.

- The tool must visualise the generated output appropriately.

- The system must be easily configurable and maintainable.

- The tool must display on request detailed information for the requested disruption.

- The system must be able to read and process CSV (comma-separated values) files.The system must be able to calculate and perform time-series analysis on the data producing list of the disruption in the network according to configurable threshold parameters (e.g. delay of more than 20 minutes in a given section of a route).

- The system must be able to update it self as near real-time as possible without user interaction or request.

- The system must be able to visualise the output as list.

- The system must be able to expand each list item and display further details.

The functional requirement require us to deal with CSV file which are currently updated each 5 minutes and pushed each 15 minutes. The technical group at TFL has agreed that it would not be problem accessing the 5 minute snapshots. However going more real-time than 5 minutes probably would be left for post deployment as it would require a formal change request to be raised which costs time and resources. The system must be capable of updating itself as near real-time as possible even though for the moment we work with 5 to 15 minutes update intervals. The CSV files that the tool would have to work contains number of fields including: âĂć Unique vehicle identifier. âĂć Time when the data has been received. âĂć The type of the bus trip (e.g. normal trip with passengers, trip to depot etc.). âĂć The route number. âĂć The last stop. âĂć Deviation from the schedule as described in the background section above. 11 âĂć Longitude and latitude. There a number of other fields which are not interesting with respect to this project. The data is unordered and there are multiple files produced one for each bus operator company. The tool should be able to aggregate all the data and analysis it and treat it as a single network.

Chapter 4

Design

This section focuses on modelling the requirements defined in the previous section in an abstract level. This allows us to better organise and structure our problem. Below I have presented initial design decisions and diagrams related to the project. This however are expected to evolve during the course of the project.

4.1 Use cases

Explanation and discussion of the use case diagram.

4.2 System architecture

Explanation and discussion of the Architecture diagram.

4.3 Class organisation

Explanation and discussion of the Class organisation. What is each class representing and what is it responsible for.

4.4 Database

Explanation and discussion of the database model.

4.5 Software Platform

Why Scala and DB Postgresql.

4.6 User Interface

Why Ruby on Rails as front end. Foundation as CSS layout framework as it is lightweight and it enables easy responsive design. Ajax short polling - as this is only prototype and is supported by all major browsers.

Chapter 5

Implementation

5.1 Data

iBus system generate very short telegram messages with the bus location [9]. This information is then processed on a server and more information is calculated and derived.

The data that has been provided by Technical Service Group (TSG) at TFL for this project consists of preprocessed iBus feeds. This feeds currently are being generated every 5 minutes. Each file consists of the following fields:

1. BusId

The information we are interested is the deviation from the schedule. This is calculated by knowing the expected arrival time of the bus at a stop from the schedule and is compared to the observed time. This value is calculated the same way for both low and high frequency buses(Low frequency buses are supposed to run according to a fixed schedule (e.g. a bus should arrive at stop at predefined time) and usually routes which have less than 5 buses an hour. High frequency bus routes should maintain headways - this meaning a bus should be arriving at stop at predefined intervals (e.g. each 2 minutes)). More about the supplied data for this project would be covered in the requirements section.

5.2 Disruption Engine

5.2.1 Input

How is the input processed etc. ...

5.2.2 Bus Network representation

5.2.3 Applying WMA

The important factors to consider are the weights and the period/window size to use. We could also apply exponential smoothing on top of the WMA. Problems: The bus could have started the journey ahead of schedule and thus to intentionally be losing time. The buses could curtail anywhere on a route without notification The buses could be diverted (this could be short term or long term) it can also be for few stops or it could be a longer diversion.

5.2.4 Applying EMA

5.2.5 Problems & Optimisations

5.3 User Interface

Chapter 6

Testing

The disruption engine consists of a number of individual components and algorithms. These are tested using a number of unit tests. Due to the nature of the engine, stress tests were carried out to test it for any memory leaks and performance issues. Evaluation of the output of the disruption engine will be performed by first carrying out further literature review to find some guidance on what window for the data to consider to use and what weights to use for optimal results. This will be followed by running the system on a given set of data (e.g. a week worth of AVL data) and comparing the output with the actual state of the network during this period.

The user interface consists of a simple web application capable of displaying list of disruptions. Testing and evaluation of this web application will be performed as user testing. As I mentioned above, some feedback and problems were identified which will be addressed and fixed. Follow up user tests will be carried out by giving access to friends and family to the web site to use and give feedback on. This should give reasonable confidence in the correctness of the user interface as there is no complex logic incorporated in the web front end application.

6.1 Unit Testing

Why it has been done and how.

6.2 Functional Testing

Why it has been done and how.

6.3 Integration Testing

Why it has been done and how.

6.4 Stress Testing

Why is it done and how it has been carried out.

Chapter 7

Results & Evaluation

7.1 Evaluation

Averaging methods: These techniques could be evaluated by calculating the error (the difference between the prediction and the actual value), the squared error and also the the sum of the squared errors (SSE) and respectively the mean of the squared errors (MSE). The model which minimizes the MSE is the best. It can be shown mathematically that the one that minimizes the MSE for a set of random data is the mean.

Consider metrics for Mean Absolute Difference and Mean Absolute Error. Peak analysis. Determining the optima Weights and the data window size.

7.2 Results

7.2.1 Problems

Data available, data frequency, no knowledge when buses do curtail, not taking into account bus dwell time, bus drivers who are running ahead of schedule could be driving slower on purpose and thus. Tool gives an upper bound of the of the WMA lost time per section.

Chapter 8

Professional & Ethical Issues

Throughout the implementation and planning stage of the project, many considerations have been taken to abide by the British Computer Society (BCS) Code of Conduct & Code of Practice [2]. These rules are introduced mainly to safeguard and protect the intellectual property of individuals who would not appreciate the use of their work by others without adequate recognition. Since unique ideas or works are in great demand, care must be taken to appreciate the works of others and to clearly reference any work that we undertake. The unauthorised use of any material in the report or the code is considered unethical. In the case of this project, a software application has been built through a combination of Open-Source libraries, code found in the internet, and my own work. This has made the progress of the project substantially faster, enabling us to adapt the work of others to better suit our requirements. Every single idea, code and piece of important information has been referenced in this report as well as the code listing, and I do not claim ownership of the work performed by others. Another important consideration I have made is to ensure the program we have developed does not provide any performance or security issues to the user's computer. One of the reasons we chose Scala as the main programming language is its automatic garbage collection. We need not worry about de-allocating memory to the program once it is not needed, as Scala runs on the Java Virtual Machine (JVM) which does this for us, enabling us to focus on more important aspects of the project. Consideration was also taken to not open up any security flaws in the system, as a result of using our software.

Chapter 9

Conclusion & Future Work

The project's conclusions should list the key things that have been learnt as a consequence of engaging in your project work. For example, "The use of overloading in C++ provides a very elegant mechanism for transparent parallelisation of sequential programs", or "The overheads of linear-time n-body algorithms makes them computationally less efficient than $O(n \log n)$ algorithms for systems with less than 100000 particles". Avoid tedious personal reflections like "I learned a lot about C++ programming...", or "Simulating colliding galaxies can be real fun...". It is common to finish the report by listing ways in which the project can be taken further. This might, for example, be a plan for turning a piece of software or hardware into a marketable product, or a set of ideas for possibly turning your project into an MPhil or PhD.

9.1 Conclusion

9.2 Future Work

It could use peak detection to make distinction between incidents and congestion. Data from other source could be used (taxis AVL, couriers services AVL) etc. Historical data could be employed in order to make further analysis and correlations with weather data, time or the day/week/year etc. Increasing the frequency of the data means that we could make use of the actual geo location information in order to calculate and monitor the bus speeds rather than the preprocessed schedule deviation value.

Using data from taxis [Cite - Requirements and Potential of GPS-based Floating Car Data for Traffic Management Stockholm Case study]

9.3 Project Retrospective

- Describe the project approach e.g. Agile incremental development approach. It pros and cons etc.
- What has worked well
- What has not worked well
- Lessons learnt

A project post-mortem, also called a project retrospective, is a process for evaluating the success (or failure) of a project's ability to meet business goals.

Post-mortems can encompass both quantitative data and qualitative data. Quantitative data include the variance between the hours estimated for a project and the actual hours incurred. Qualitative data will often include stakeholder satisfaction, end-user satisfaction, team satisfaction, potential re usability and perceived quality of end-deliverables.

References

[1] Greater London Authority. Number of buses by type of bus in london. http://data.london.gov.uk/dataset/number-buses-type-bus-london, 2014. Online; accessed 22-October-2014.

[2] BCS. Bcs code of conduct. http://www.bcs.org/category/6030, June 2011. Online; accessed 6-April-2015.

[3] Bowen T. Clarke R. and Head J. Mass deployment of bus priority using real-time passenger information systems in london. pages 891–921, 2007.

[4] Transport for London. Transport for london's ibus wins innovation award. https://www.tfl.gov.uk/info-for/media/press-releases/2008/march/transport-for-londons-ibus-wins-innovation-award, March 2008. Online; accessed 2-April-2015.

[5] Transport for London. https://www.tfl.gov.uk/cdn/static/cms/documents/uploads/forms/lbsl-tendering-and-contracting.pdf, 2008. Online; accessed 2-April-2015.

[6] Transport for London. All london's buses now fitted with ibus. https://www.tfl.gov.uk/info-for/media/press-releases/2009/april/all-londons-buses-now-fitted-with-ibus, April 2009. Online; accessed 2-April-2015.

[7] Transport for London. https://www.tfl.gov.uk/info-for/media/press-releases/2009/may/

`centrecomm-celebrates-30-years-keeping-londons-buses-moving`. Online, May 2009. Online; accessed 2-December-2014.

[8] Transport for London Media. `https://www.tfl.gov.uk/info-for/media/press-releases/2014/may/annual-passenger-journeys-on-london-s-buses-top-2-4-billion`. Online, May 2014. Online; accessed 2-December-2014.

[9] N.B. Hounsell, B.P. Shrestha, and A. Wong. Data management and applications in a world-leading bus fleet. *Transportation Research Part C: Emerging Technologies*, 22(0):76 – 87, 2012.

[10] Metin ZontulT Mehmet Altinkaya. Urban bus arrival time prediction: A review of computational models. *International Journal of Recent Technology and Engineering (IJRTE)*, 2(4), September 2013.

[11] BBC News. Extra 500 buses planned for growing capital before 2021. `http://www.bbc.co.uk/news/uk-england-london-30285777`, 2014. Online; accessed 2-December-2014.

[12] MINA BASIRAT SEYED MOJTABA TAFAGHOD SADAT ZADEH, TONI ANWAR. A survey on application of artificial intelligence for bus arrival time prediction. *Journal of Theoretical and Applied Information Technology (JATIT)*, 46(1):516 – 525, December 2012.

[13] Alan Wong and Nick Hounsell. Using the ibus system to provide improved public transport information and applications for london. Paper 01753, July 2010.

Appendix A

Extra Information

A.1 Use Case Diagram

A.2 Architecture Diagram

A.3 Class Diagram

A.4 Relational Model

Appendix B

User Guide

You must provide an adequate user guide for your software. The guide should provide easily understood instructions on how to use your software. A particularly useful approach is to treat the user guide as a walk-through of a typical session, or set of sessions, which collectively display all of the features of your package. Technical details of how the package works are rarely required. Keep the guide concise and simple. The extensive use of diagrams, illustrating the package in action, can often be particularly helpful. The user guide is sometimes included as a chapter in the main body of the report, but is often better included in an appendix to the main report.

B.1 Installation

B.2 Execution

B.3 Dependencies

Appendix C

Source Code

Complete source code listings must be submitted as an appendix to the report. The project source codes are usually spread out over several files/units. You should try to help the reader to navigate through your source code by providing a "table of contents" (titles of these files/units and one line descriptions). The first page of the program listings folder must contain the following statement certifying the work as your own: "I verify that I am the sole author of the programs contained in this folder, except where explicitly stated to the contrary". Your (typed) signature and the date should follow this statement.

All work on programs must stop once the code is submitted. You are required to keep safely several copies of this version of the program - one copy must be kept on the departmental disk space - and you must use one of these copies in the project examination. Your examiners may ask to see the last-modified dates of your program files, and may ask you to demonstrate that the program files you use in the project examination are identical to the program files you had stored on the departmental disk space before you submitted the project. Any attempt to demonstrate code that is not included in your submitted source listings is an attempt to cheat; any such attempt will be reported to the KCL Misconduct Committee.

You may find it easier to firstly generate a PDF of your source code using a text editor and then merge it to the end of your report.

There are many free tools available that allow you to merge PDF files.

C.1 Engine

C.2 Web Front End

C.3 Database

C.4 Dependencies

Scala and Java versions, scalatest library, PostgreSQL, scala xml library, scala geo coordinate library, scala logger Ruby on Rails, Ruby version, Gems that have been used Foundation, jQuery - Include the specific versions