RC25258 (D1201-003) January 10, 2012
Computer Science

IBM Research Report

System and Analytics for Continuously Assessing Transport Systems from Sparse and Noisy Observations: Case Study in Dublin

L. Gasparini, E. Bouillet, F. Calabrese, O. Verscheure
IBM Research
Smarter Cities Technology Centre
Mulhuddart
Dublin 15, Ireland

Brendan O'Brien, Maggie O'Donnell
Dublin City Council
Road Administration
Ireland

Research Division
Almaden - Austin - Beijing - Cambridge - Haifa - India - T. J. Watson - Tokyo - Zurich

System and Analytics for Continuously Assessing Transport Systems from Sparse and Noisy Observations: Case Study in Dublin

L. Gasparini, E. Bouillet, F. Calabrese, O. Verscheure[1]
Brendan O'Brien, Maggie O'Donnell[2]

***Abstract*— This paper describes our implementation of an Intelligent Transportation System, built for the road administration authority at Dublin City Council. The system is able to ingest data from different sources, while providing updated speed and traffic flow measurements, travel time estimates and statistical aggregations of current traffic conditions in real-time to the user. The application is powered by IBM Infosphere Streams, which enables it to easily scale to hundreds of thousands of data points processed per second, while maintaining sub-second delay from data acquisition to delivery. In this work we provide an overview of the analytics and the application architecture used to deliver the required capabilities.**

I. Introduction

The first decade of this millennium marks a watershed in the history of urban expansion, when the human population grew more urban than rural. The proportion of human population rose from 13 percent in the 1900s to 49 percent in 2005, and is projected to reach 60 percent by 2030. By 2030 the urban population is expected to approach 5 billions dwellers, a 43% increases from today's 3.5 billions. The rapid growth of demand for transportation and high levels of car dependency caused by urban sprawl has resulted in severe traffic congestion, associated productivity loss, and environmental degradation in many areas. Adding capacity through construction of new facilities within the city landscape is a very difficult endeavor due to space constraints and prohibitive costs. The general consensus is that congestion reduction is instead better addressed through Intelligence Transportation Systems (ITS) that leverage sensor networks, communications and computing technologies to manage existing infrastructure and transportation systems more efficiently. Intelligent Transport Systems are used to gain an insight about why and how people and goods flow across the city. They are also used to assist road operators with incident detection and help them to quickly react to resulting traffic pattern changes and minimize their impacts. An important development within ITS is the emergence and installation of sensor technologies for collecting data on the state of the transport system [1]. Examples of this are Automated Vehicle Location (AVL) systems or other opportunistic sources of trajectory data including smartphones, CCTV systems with license plate recognition capability, and induction loops at traffic intersections such as the ones used in Sydney Coordinated Adaptive Traffic Systems (SCATS®), or Split Cycle Offset Optimization Technique (SCOOT). Intelligent Transportation Systems can make use of such data collected from a variety of sources to enable real time traffic monitoring and management. They also offer a mean to increase the public awareness about more energy efficient and sustainable modes of transportation e.g. public transit. Awareness which when combined with real-time information, such as expected arrival times, contribute to minimize user uncertainties about their travel plans, improve user experience, foster adoption of sustainable modes of transport, and as a whole a more efficient usage of the infrastructure.

There are, however, major challenges in building systems that are flexible and powerful enough to handle diverse demands from a large user base. The first challenge is one of scalability. As various kinds of sensor technologies become ubiquitous, the massive amount of data they produce must be fused and analyzed, often in real-time. The processing rate of such data for a city wide system can easily exceed millions of measurement points per second. In addition, data must be integrated in large system states (e.g., road networks) that can potentially contain millions of elements. A second challenge is the quality of the data. As the sensor technologies grow larger so does the risks of receiving invalid data from faulty sensors, or noisy data from less accurate sensors, such as GPS sensors. Data sparsity is another challenge. Even though the volume of data can be extremely large, because of the equally large dimension space of the system, the volume is usually insufficient for deriving accurate traffic models. A final challenge is the development of the computing infrastructure required to support the needed functionality of ITS, especially given the large volumes and variety of data available and the diverse set of parties involved in providing this data, such as government agencies, commercial enterprises, legacy systems, and end-user commuters. Studies have shown that developing and integrating the various components of an ITS infrastructure constitute a significant portion of the capital cost and complexity of such systems. These systems access a broad spectrum of data source types, that produce a heterogeneous mix of content with varying degrees of quality. The different types may necessitate different kinds of software components to process the data. Also, the systems developed by the different parties are not necessarily developed with interoperability in mind. These factors add to the complexity of the system. Furthermore, different kinds of end-users have different needs for the traffic-data. These end-users include commuters, highway patrols, emergency

[1]Authors are with the IBM Dublin Research Laboratory, Ireland [luca.gasparini|bouillet|fcalabre|oliversc]@ie.ibm.com

[2]Authors are with the Dublin City Council, Road Administration, Ireland

vehicles, departments of transportation, urban planners, commercial fleet operators, etc. These users not only pose large numbers of simultaneous analysis requests, but also require analyses of significantly different natures. For example, the department of transportation requires real time processing of detailed traffic data across the urban area to perform dynamic traffic management, whereas, the analysis performed by urban planners rely on high level aggregation of the data from historical databases. This further increases the complexity of the system.

In this paper, we propose a generic framework upon which an Intelligent Transport System can be built that tackles the challenge mentioned above. The framework is based on the IBM InfoSphere Streams platform [2], a scalable stream processing platform. We describe a case study demonstrating the use of our framework and InfoSphere Streams for Intelligent Transportation Systems with an actual application implemented for the road administration authority at the Dublin City Council to manage the public transport.

The paper is structured as follows: In the rest of the paper, we describe the InfoSphere Streams platform and our case study in the Intelligent Transportation Systems space. In section 2 we review existing work in public transit ITS. In section 3 we provide our vision of the platform and the core elements that constitute it. In section 4 we introduces InfoSphere Streams platform and its features. Section 5 describes the implementation of the case study for the Dublin City Council and Section 6 provides some experimental results for our system. We then describe future work in Section 7 and finally conclude.

II. State of the art

Systems for monitoring and managing public transportation have been one of the first applications of ITS. They have been used to plan public transportation routes and schedules, and are increasingly used to allow for real time management of the fleet by collecting data about the real time position of vehicles.

They can for instance be used for planning bus routes and schedules, in order to evaluate and equalize accessibility levels or different areas of the city [3]. By monitoring the real time position of vehicles and comparing it with the scheduled ones, supervision strategies can be implemented to improve service reliability [4], including holding strategy to keep headway in bus networks [5]. To this end, several research has been developed for the prediction of arrival times of vehicles (see [6]–[8] and references therein) as function of current and historical traffic conditions.

On the other hand, information about functioning of the public transport system is increasingly seen as key element for citizens to decide whether to use public transit. For this reason, department of transportations have started to provide public transportation schedules information to be used for end-users applications. To facilitate this, Google defined a specification for data format called General Transit Feed Specification (GTFS) which is now well adopted as common format for public transportation schedules and associated geographic information, used as input in many web based and mobile phone applications. Recently, this information is being complemented with real time passenger information (RTPI) such as predicted arrival times of buses at a bus stop, to further improve public transit functioning awareness [9]. This information is being provided using several user interfaces such as displays at bus stops as well as mobile phone applications, which have shown a concrete impact on how users perceive the reliability of the public transport system [10]. Of crucial important for this new information to be effective is the reliability of the predictions.

In this paper we present a system we are developing that allows supporting both public transit management as well as RTPI applications. It is based on the assimilation and processing of real time vehicle locations to

- Estimate traffic condition on the street network by fusing measurements from fixed (e.g. loop detectors) and mobile (e.g. vehicles) sensors;
- Evaluate anomalies in the public transportation system;
- Predict time of arrival of vehicles from the analysis of traffic conditions as measured by all available sensors.

III. The vision

We envision a system that is able to process real-time Automatic Vehicle Location data, generate different kinds of real-time traffic statistics, and perform customized analyses to answer a variety of user requests such us anomalies detection. Examples of customized analyses include continuously updated speed and traffic flow measurements for all the different bus lines in the city, estimate travel times between points along the public transport routes, statistical aggregations of current traffic conditions and real time prediction of arrival times of public transport vehicles at their next stops along their route.

The system we are developing adopts a streaming computing platform [2] which is able to perform complex analytics on extremely high volumes of heterogeneous data with very low latency performances. This enables services that would otherwise be thought hardly possible with affordable computational power, such as multimodal dynamic journey advisors that assist travelers before and during their journey. Instead of basing their decision on static or semi-static information, dynamic journey advisors continuously update their decisions as new data is received from a variety of sources, and keep the traveller informed on their GPS equipped smartphone during their journey. Examples of notifications include arrival at their destination or a connection (assuming they opt to be tracked), or incidents that could impact their trip, such as a missed connection if the system predicts a delay on their current route. The update can include reports about the incident, possibly illustrated with crowd-sourcing information, and alternate routes if available.

Figure 1 depicts the main constituents of our Intelligent Transport System application. The application consists of two main processing components. The first is an offline process, shown at the top of the figure. This process extracts a profile of the transport infrastructure from historical AVL

trace data, such as the physical topology of the public transport network and a predictive model of travel times. The offline profile is in turn used by the other component of the application, which ingests live AVL data, and possibly other data sources supported by the predictive model, to compute:

- for the road administrator, the key performance indicators needed to monitor the health of the transport system, and
- for the end-user the estimated waiting times as well as suggested routes to reach their destination, which can be updated as more information becomes available.

The next subsections describe both components in details.

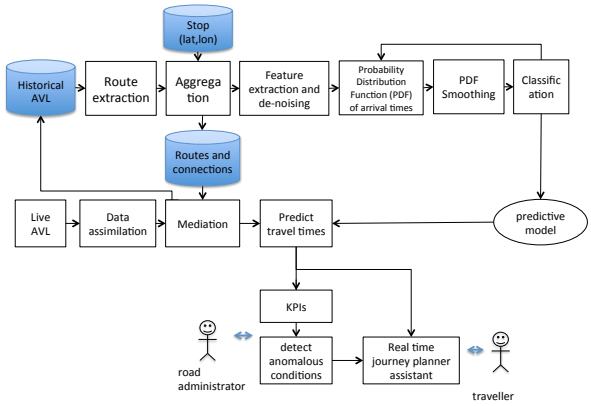

Fig. 1. Intelligent Transport Systems application flow graph.

A. Offline processing

The offline process serves a number of purposes, one of which is topology extraction of the transportation network. We consider here both the logical and the physical graph representations commonly used to describe transport network topologies. In a logical graph representation, nodes correspond to bus-stops (or train-stations) and edges are used to indicate pairs of consecutive stops traversed by bus lines. This representation is often used in offline journey planning. However, dynamic journey planners that base their decision on realtime traffic observations must complement this view with a physical view that describes the geographical features of the logical edges. Knowledge of the physical infrastructure is important because it can capture traffic condition dependencies that may not appear in the logical view, such as road segments shared between bus lines that do not share bus-stops. Most systems make use of the logical view, while the physical view is either absent or outdated. This is often because this information is not always provided by the operator. In the offline process we thus include a component which uses historical GPS data to compute this view automatically. This component includes a stop-localization feature which measures the density distributions of the GPS measurements to locate the scheduled stops along their respective routes. The feature extraction operator measures the time of departure of the k^{th} bus since the start of the day at every stop, and individual travel times between consecutive stops. The offline process then aggregates the extracted features by time of day and day of weeks and computes their Probability Distribution Functions (PDF). Because the data is usually sparse it further smooths the PDFs using either a Max Likelihood Estimation, or Kernel Density Estimation. Finally we classify the PDFs depending on the time of day or day of week, or type of day and weather using clustering or decision methods. The results of this classification is our predictive model for the time of arrivals at the bus stops. Note that the application flow-graph could be augmented to include other sources of the data in the classification, such as induction loop data.

B. Online processing

The predictive model is in turn used by the other component of the application, which ingests live AVL data, and possibly data from other sources used by the travel-time model classifier. The online component uses the live observation data to predict arrival times. For the road administrator it computes the key performance indicators needed to monitor the health of the transport system. For the end-user it uses the predicted arrival time information from the various modes of transport which it combines with other forms of static data (such as walking or biking distances), into a stochastic optimization process. The optimization provides the traveller with a list of possible choices and level of confidence to reach a desired destination from their current location within specified constraints, such as maximum time of arrival, or maximum waiting time at connections.

IV. The Stream Processing Platform

Applications in IBM InfoSphere Streams take the form of graphs of modular, reusable software components (called operators) interconnected by data streams. Each operator ingests fine-grained data of certain content and format and performs various analysis, the result of which can be used in downstream operators. These applications are deployed on a distributed runtime infrastructure to allow scalability via pipelining and parallelization. InfoSphere Streams provides various services to manage the infrastructure and the applications deployed on it so as to support high-throughput, low-latency stream processing. One of the key features of InfoSphere Streams is its component-based programming model. This allows composing and reconfiguring individual components to create different application flow-graphs that perform different kinds of analysis. In addition, it enables the creation and deployment of new applications without disrupting existing ones. Furthermore, it allows the new applications to reuse intermediate derived streams produced by existing applications in order to minimize duplicate or redundant processing of data. This facilitates the growth and incremental incorporation of new analytics, and help in tackling the challenges of dealing with diverse data sources and diverse end-user needs.

V. Application Implementation

The implementation of the public transit monitoring application is based on a modular design, with two different

modules handling different aspects of the problem. At the lower level, an InfoSphere Streams instance provides all the analytics used by our application, illustrated in Figure 2, and handles the interaction with the data sources; on the other end, a Java-based web server is used to deliver the information up to the user (public transit operator), in the form of both HTML pages and KML data. The real-time bus location data is provided to our application using the SIRI protocol (Service Interface for Real Time Information), a CEN standard widely used throughout Europe. A specific adapter component is used to subscribe to the SIRI server and collect data from all the vehicles equipped with GPS receivers. The component (green in Figure 2), consists of a light weight HTTP service operator paired with a XML parser to convert an infinite stream of HTTP POST chunks into a continuous stream of GPS probe records. Once inside the system, the data passes through different processing blocks that take care of:

- Mediation and de-noise the GPS signals (blue in Figure 2); This component removes expired records and out-of-bound records. It maps the vehicles on a route with a confidence interval that increases with the number of GPS observations. The component estimates the vehicle's speed and direction along the route, smoothed in space and time in order to remove noise. This information is used to identify vehicles that advertise an inconsistent route number or destination (typically while the vehicle is out of service) so that they are removed from the estimation of arrival times;
- Estimate the average speed and travel time for every segment of the network; This component (light-red in Figure 2) computes the travel time per distance units δ_t/δ_x (in s/m) along the displacement $\delta_x = x_{i+1} - x_i$ between two positions x_i and x_{i+1} of a vehicle along its assigned route, as shown in Figure 3. The metrics is then used to update the travel time t_{uv} of every intermediate segment (u, v) traversed by the vehicle from x_i to x_{i+1}. We use the Exponentially Weighted Moving Average $t_{uv,j+1} = (1 - \alpha_t)d_{uv}\delta_t/\delta_x + \alpha_t t_{uv,j}$, where d_{uv} is the length of the segment, and α_t is a decay factor that decreases from 0.5 to 0 as the time interval between two samples j and $j+1$ affecting segment (u, v) increases. The decay of parameter α ensures that the effect of the observations are weighted according to their actual age rather than order of arrival only. When travel times are updated on route segments, the estimated times of arrival at stops along the routes that share the segments are also updated.
- Predict the arrival time of the vehicles on the different bus stops; The vehicle positions and the travel times along the segments of their respective routes are used to estimate the times of arrival of the vehicles at their stops ahead. The estimated times of arrivals are immediately updated when the travel times t_{uv} on the route segments or the position of the vehicles are updated. If a vehicle position is not received by the time it is expected to arrive at a stop, it is automatically removed from the list of vehicle scheduled for that stop by assuming that it has maintained its last known averaged speed.

Other components in the application compute the system level key performance indicators, such as number of received valid and invalid records per second (black in Figure 2), and per-vehicle measurements, such as vehicle bearing, averaged speed, and frequency of updates (dark-red in Figure 2).

The information is then sent to the Java web server that handles the presentation. Our application makes use of KML 2.2 to represent geographical information, which could be in turn rendered by any compatible client (tests were conducted using Google Earth 5.2). Since KML does not define a way to capture user input, we had to find a viable way to handle this type of interaction. Our solution is then represented in 4: we maintain a server-side replica of the elements displayed on the client, that the user can alter by interacting on a specifically crafted web site; this replica is then synced back to the client using KML NetworkLinks and NetworkLinkControls, in a way that only the actual updates relevant to the visible region will be transmitted.

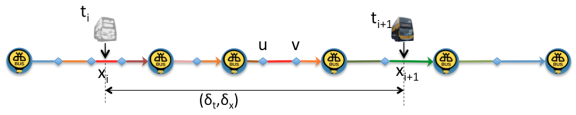

Fig. 3. Estimation of average speed for every segment of road network.

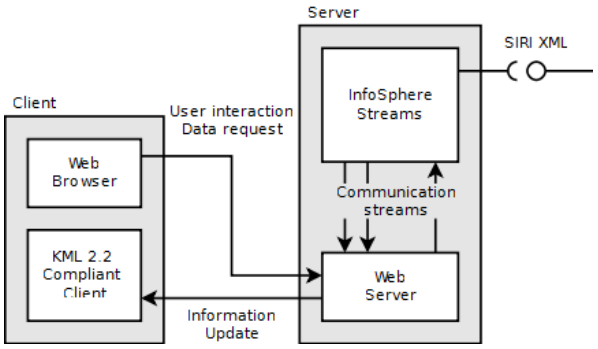

Fig. 4. Application architecture.

In Figure 5 we show a screenshot of the running application. On the web interface, it is possible to note the search field, the selected elements list and the controls to add or show on the map elements related to the current one; the routes, stops and buses are instead shown on the map. A video of the running application is available at [11]

VI. Experimental results

We deployed the application at the Dublin City Council and tested it with live SIRI data from the Dublin Bus operator. The hosting server consists of 8 Intel(R) Xeon(R) X5650 2.67GHz cores and 6GBytes of memory running RHEL 5.3 64bits in a virtualized ESX VMware(R) environment. The bus network consists of 4691 stops, interconnected by 631 bus routes (including route variations on the same bus line).

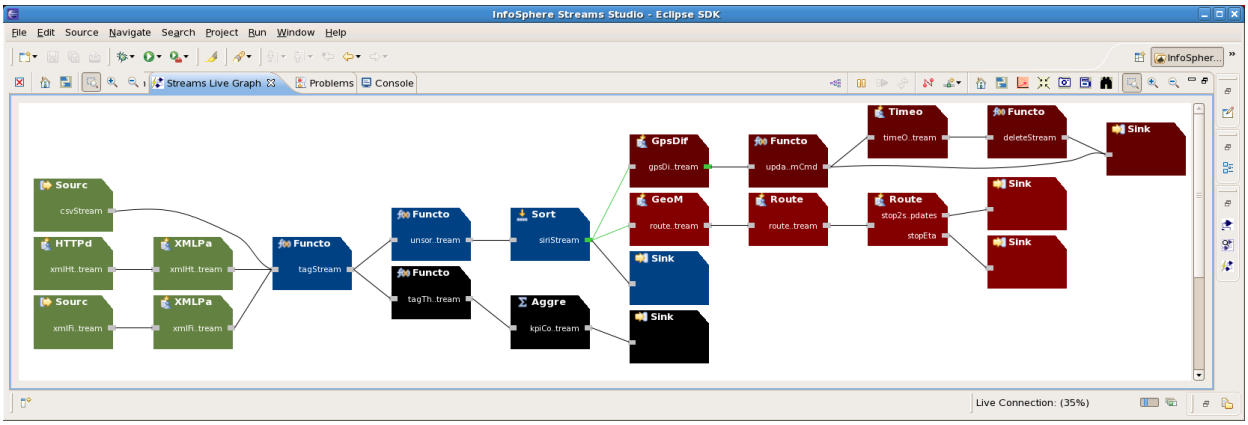

Fig. 2. Stream processing application flowgraph screenshot.

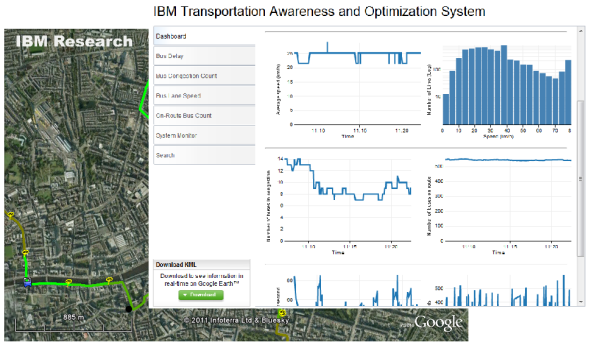

Fig. 5. Application screenshot.

The geographical features of the road map data used to represent the bus routes consist of about 725 thousands map elements. The fleet managed by the operator consists of about 1000 vehicles. Each vehicle sends its GPS position every 20 second on the average. This rate does not include inaccurate, expired, or incomplete records, and records from vehicles not assigned to a bus line, all of which are ignored by our application. Figures 6 and 7 show the respective distributions of time intervals and travelled distances between GPS records. Note that the bus are at a stop and not traveling (i.e. distance is 0m) for a good part of the time. The application receives on the average a total of 50 records per second during the peak time of the day, when most of the fleet is in circulation. After removing vehicles that are not in service early-on in the application the flow thins down to about 25 data points per second, as illustrated in Figure 8. The application achieves sub-second latencies from the time a record is received to the moment the state is updated. The application analytics use 2.5% of CPU on the average and 33% of memory. This distribution of resource consumption is emblematic of the underlying streams programming paradigm used in our application. This is because the application state is stored in memory and thus the analytics avoid the overhead of accessing this information from a database. The statistical observations of the performance metrics measured in real-time for each vehicle and along every sections of the bus routes account for 90% of the memory footprint. Note that the runtime environment can easily distribute this state across multiple servers if needed.

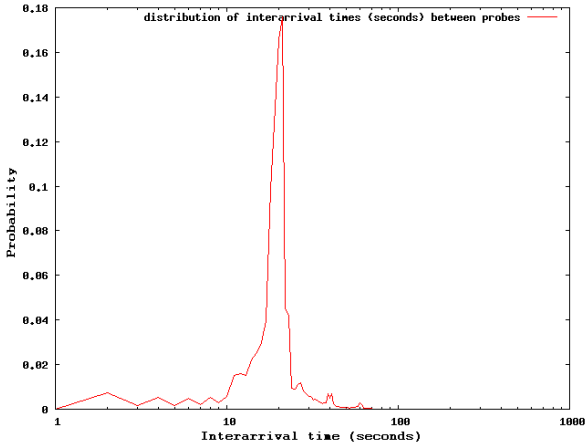

Fig. 6. Distribution of time intervals between two consecutive position records per vehicle (active vehicles only).

VII. Future Work

The choice of aggregating data primarily at road network link level, instead of at bus route level, enables the system to react to a number of different data sources, some of which are not yet explored. The first and most important additional source comes from SCATS data, produced by a series of induction loops positioned beneath the road surface, and able to estimate the vehicle rate and the distance between them; the loops are generally deployed in the main intersections of the city, and in the specific case of Dublin are present in more than 600 intersections. This data could be used to identify congestion situations, as well as validating the information received from the GPS sensors. Weather historical data and forecasts could be used as well to correct incoming records; these are in fact biased by road and visibility conditions, and being able to offset this bias may lead to non-trivial improvement in the travel time estimates. Another possible source of data comes from mobile phones, that are now pervasive

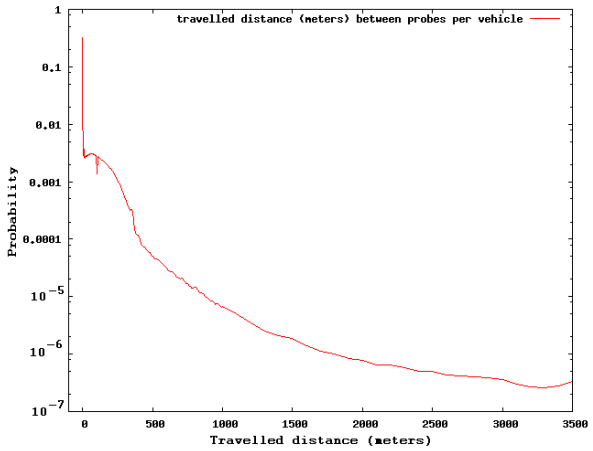

Fig. 7. Distribution of travelled distances between two consecutive position records per vehicle (active vehicle only).

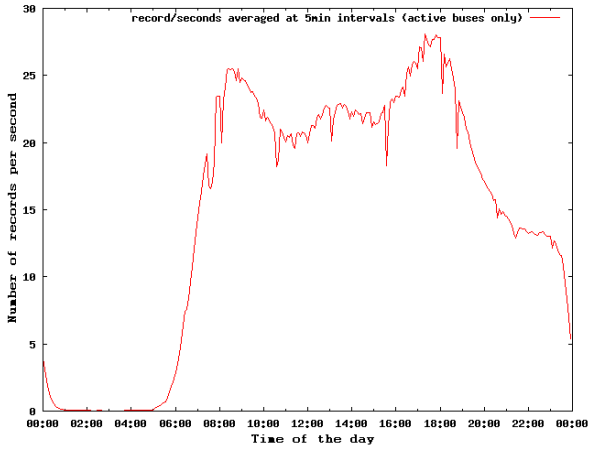

Fig. 8. Number of records (active vehicle only) received during the day, averaged at 5mins intervals.

throughout the population [12]. However, the effective use of this information is made difficult by issues like accuracy, privacy and delay needed for the data to reach the server. While the use of more diverse data may possibly lead to improvements in prediction accuracy, directly improving the prediction algorithm is another viable and promising path. In particular, we are looking forward to build a continuous model of the travel speed during the day for each link on the network; this model will contain information about expected value and variability of the travel speeds in different hours of the day, for every class of weekday (festive, pre-festive and working day). We will then be able to use it to better average the speed recorded on the field, and to predict the travel times that are likely to be seen in the future.

VIII. Conclusions

In this work we presented our framework to implement an Intelligent Transport System, currently deployed at the road administration authority of Dublin City Council. The system is able to ingest data from different sources, while providing updated speed and traffic flow measurements, travel time estimates and statistical aggregations of current traffic conditions in real-time to the user. The framework is built on top of IBM Infosphere Streams, an efficient and scalable environment for building streaming applications. We presented an overview of the analytics included, which handle all the required processing steps, from data mediation and de-noising up to customized analyses based on user requests. With experiments we demonstrated the performances of our implementation, able to process and display the AVL data of individual buses from a fleet consisting of over a thousand vehicles using a single server with four x86 CPU cores used at less than 1% per core. As a whole, the presented system permits to overtake the challenges of scalability, complexity and data diversity that ITS systems face nowadays.

IX. Acknowledgments

The authors gratefully acknowledge the Dublin City Council for providing the access to the bus data.

References

[1] C. Antoniou, R. Balakrishna, and H. Koutsopoulos, "Exploiting emerging data collection technologies for dynamic traffic management applications," *Proceedings of the 12th World Conference on Transport Research (WCTR)*, July 2010.

[2] A. Biem, E. Bouillet, H. Feng, A. Ranganathan, A. Riabov, O. Verscheure, H. N. Koutsopoulos, and C. Moran, "Ibm infosphere streams for scalable, real-time, intelligent transportation services," in *SIGMOD Conference*, 2010, pp. 1093–1104.

[3] F. Pinelli, A. Hou, F. Calabrese, M. Nanni, C. Zegras, and C. Ratti, "Space and time-dependant bus accessibility: a case study in rome," in *IEEE International Conference on Intelligent Transportation Systems*, Sr. Louis, Mussouri, USA, October 2009.

[4] C. Pangilian, N. Wilson, and A. Moore, "Bus supervision deployment strategies and use of real-time automatic vehicle location for improved bus service reliability," *Transportation Research Record: Journal of the Transportation Research Board*, pp. 28–33, 2008.

[5] B. Yu and Z. Yang, "A dynamic holding strategy in public transit systems with real-time information," *Applied Intelligence*, vol. 31, pp. 69–80, 2009.

[6] D. Tiesyte and C. S. Jensen, "Similarity-based prediction of travel times for vehicles traveling on known routes," in *Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems*, ser. GIS '08. New York, NY, USA: ACM, 2008, pp. 14:1–14:10.

[7] ——, "Assessing the predictability of scheduled-vehicle travel times," in *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, ser. GIS '09. New York, NY, USA: ACM, 2009, pp. 416–419.

[8] R. Padmanaban, K. Divakar, L. Vanajakshi, and S. Subramanian, "Development of a real-time bus arrival prediction system for indian traffic conditions," *Intelligent Transport Systems, IET*, 2010.

[9] B. Taylor, H. Iseki, M. Miller, and M. Smart, "Thinking outside the bus: Understanding user perceptions of waiting and transferring in order to increase transit use," California PATH Research Report UCB-ITS-PRR-2009-8, Tech. Rep., 2009.

[10] B. Ferris, K. Watkins, and A. Borning, "Location-aware tools for improving public transit usability," *IEEE Pervasive Computing*, vol. 9, pp. 13–19, 2010.

[11] IBM, "Dublin application." [Online]. Available: http://www.youtube.com/watch?v=VBv5XRGQ7nA

[12] H. Wang, F. Calabrese, G. DiLorenzo, and C. Ratti, "Transportation mode inference from anonymized and aggregated mobile phone call detail records," in *IEEE International Conference on Intelligent Transportation Systems*, Madeira Island, Portugal, September 2010.