Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 4, Issue. 7, July 2015, pg.88 – 92

RESEARCH ARTICLE

REAL TIME PUBLIC TRANSPORT INFORMATION SERVICE

Ch. Lavanya
M.tech (Embedded Systems)
lavanya.chekurthi363@gmail.com
Jits
karimnagar

B. Sreenivas
Assoc.Prof
Targetsrinu8@gmail.com
Jits
karimnagar

Abstract—Modern cities continuously struggle with infrastructural problems especially when the population is massively grow-ing. One affected area is public transportation. In default of offer-ing convenient and reliable service the passengers tend to consider other transport alternatives. However, even a relatively simple functional enhancement, such as providing real-time timetable information, requires considerable investment and effort following traditional means, e.g. deploying sensors and building a background communication and processing infrastructure. Using the power of crowd to gather the required data, share information and send feedback is a viable and cost effective alternative. In this demonstration, we present TrafficInfo, our prototype smart phone application to implement a participatory sensing based live public transport information service. TrafficInfo visualizes the actual position of public transport vehicles with live updates on a map, and gives support to crowd sourced data collection and passenger feedback.

Keywords—Participatory sensing, Crowdsensing, Public trans-port, GTFS, Publish/subscribe, XMPP, Demonstration.

1. INTRODUCTION

Most of today's modern cities come up with different smart services from time to time to make the daily routines of their inhabitants more convenient. However, implementing smart city services traditionally requires considerable infrastructural investments which are expensive, resource intensive and time consuming. For instance, as public transportation is one of the most used city services in modern cities its continuous maintenance and improvement is inevitable. Nevertheless, introducing even a relatively simple extension to the basic service, like a timetable updated in real-time, calls for the deployment of a costly vehicle tracking infrastructure (consisting of sensors; communication and back-end informatics system; front-end user devices; etc.).

An alternative approach is to exploit participatory sensing or often called mobile crowdsensing [1], which does not need such investments, rather it is based on the power of crowd to gather live data. Fig. 1 shows this scenario. In this case, the built-in sensors of the passengers' mobile devices, or the passengers themselves via reporting incidents, provide the required data for vehicle tracking and send instant route information to the service provider. The service provider than aggregates, cleans, analyzes the collected data, and derives and disseminates the real-time updates. For sensing the built-in and ubiquitous sensors of the mobile phones are used either in participatory or opportunistic way depending on whether data collection happens with or without user involvement. The contribution of every traveler can be useful. Hence, passengers waiting for a trip can report the line number with a timestamp of every arriving public transport vehicle at a stop during the waiting period. On the other hand, on board passengers can send actual position information of the moving vehicle periodically and report the event of arrival at/departure from a stop.

In this demo, we introduce and demonstrate our participatory sensing based TrafficInfo application. TrafficInfo is a simple and easy-to-use Android application which visualizes real-time public transport information of the given city on top of Google Maps. It is built upon our XMPP (Extensible Messaging and Presence Protocol) [2] based communication framework [3] for developing crowd assisted smart city applications. Following the publish/subscribe (pub/sub) communication model the passengers subscribe in TrafficInfo, according to their interest, to traffic information channels dedicated to different public transport lines or stops. Hence, they are informed about the live public transport situation, such as the actual vehicle positions, deviation from the static timetable, crowdedness information, etc.

To motivate user participation in data collection we offer a day zero service to the passengers, which is a static timetable. It is built on GTFS (General Transit Feed Specification) [4] based transit schedule data provided by public transport opera-tors. TrafficInfo basically collects the timestamped halt events of the public transport vehicles at the stops and/or simple annotation data, such as report on crowdedness or damage.
This requires at the moment user interaction but we have been working on an automated event detection method, as well. Our experimental system was designed to gain insight into the exciting area of crowdsensing based applications via real field experiments. The main contribution of our work is a proof-of-concept implementation of a real world application scenario which opens the way for more thorough investigations and can reveal open issues which were never thought.

2. PUBLIC TRANSPORT INFORMATION SERVICE

TrafficInfo basically visualizes public transport vehicle movements on a map. When real-time data, collected by the passengers, is available the position information gained from the static timetable is refreshed reflecting the actual traffic situation. Thus, passengers can have an idea whether the selected vehicle is on time or delayed and they have to hurry or look for some alternative transport. Beyond gathering timestamped data of halt events at the stops social interaction is also supported in TrafficInfo through sharing useful information for the passenger community, like crowdedness, controller on board, malfunctions or damages, etc.

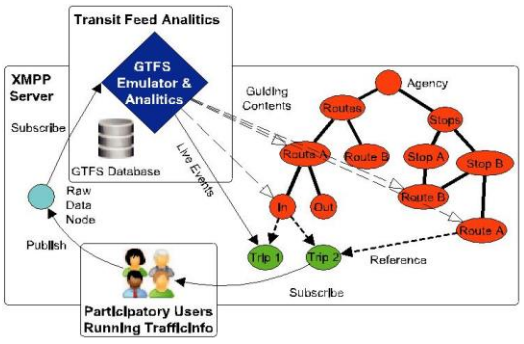

Fig : 2: Demonstration System Architecture

A. Demo Setup

Our demonstration system architecture (see Fig. 2) consists of a standard XMPP server, a GTFS Emulator with an analytics module, and the TrafficInfo application.

Our live public transport information service fits well with the asynchronous publish/subscribe communication model where members of the crowd collect and publish data, and passengers subscribe to live information channels. Thus, the XMPP server maps the public transport lines to a hierarchical pub/sub channel structure. We turned the GTFS database into an XMPP pub/sub node hierarchy. This node structure facilitates searching and selecting transit feeds according to user interest. The pub/sub node model for content filtering in a transport information feed is depicted in Fig. 2.Hierarchy contain only persistent data and references relevant to the trips. The users can access the transit data via two ways, based on routes or stops. When the user wants to see a given trip (vehicle) related traffic information the route based filtering is applied. On the other hand, when the forthcoming arrivals at a given stop (location) are of interest the stop based filtering is the appropriate access way.

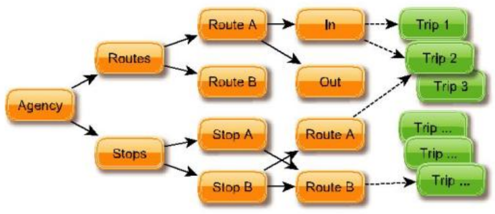

Fig. 3. Publish-Subscribe Model for GTFS Feeds

The GTFS Emulator provides the static timetable information as the initial service. It uses the officially distributed GTFS database of the Center for Budapest Transport. The analytics module is in charge of the business logic offered by the service, e.g., deriving crowdedness information or estimating the time of arrivals at the stops from the data collected by the crowd.

TrafficInfo handles the subscription to the pub/sub channels, publishes events to and receives updates from the XMPP server, and visualizes the received information.

B. Demo Requirements

The only on-site requirement for our demonstration is Wi-Fi based Internet access. The XMPP server and the GTFS Emulator with the analytics module will run at our remote premises and will be accessed via the Internet. On-site we will use in parallel multiple Android devices equipped with TrafficInfo to send/receive massages to/from the XMPP server, and show as the live traffic updates appear in real-time on the devices. Attendees can interact with the application and send information, e.g., ’faked’ crowdedness data, which will be disseminated in real-time and appear at once on the other devices.

C. TrafficInfo Details

After launching TrafficInfo the application connects to the XMPP server. The opening screen is the Google Map of the city in question on top of which the public transport information is visualized. In our case, this city is Budapest, the capital of Hungary. The user can follow the actual positions of the vehicles associated with the pub/sub channels the user has been subscribed to (see Fig. 4). Channel subscription can be initiated and managed under the Settings menu. In the News panel, annotations are listed containing information about the active vehicles. Visualizing the transit feeds and traffic updates in an appropriate manner has a primary importance as it serves the ultimate goal of TrafficInfo. Thus, clickable pictograms illustrate on the map the vehicles associated with the sub-scribed traffic channels. The user can navigate to a reporting board by clicking on the piktogram where he has several options to send feedback, such as crowdedness indication, report on damage or free text message. The server side processing logic in the analytics module of this feedback data is a challenging issue which has been under continuous development.

D. Data Gathering

Crowd sourced services are based on the data collected by the users. In general, the more data is gathered the higher is the Quality of Experience (QoE) of the service. However, it is important to know what the relevant data is. In our case, data is needed to identify line information, vehicle position, halt events at the stops with high accuracy, and to detect transportation events, such as traffic jams, malfunctions and unusual things. It is critical to be able to automatically detect these events, but at the moment event detection is possible only via user interaction in our application.

Towards this end, we introduced a sensor data measurement module in TrafficInfo which measures and logs continuously the actual values of the built-in sensors of today's Android based smart phones. The automatically logged data is: i) Base Station information; ii) Wi-Fi network details; iii) accelerometer information; and iv) GPS data. Thus, the implemented network event listeners automatically monitors mobile cell transitions, hand-overs and cellular network parameter changes, neighboring cell information, Wi-Fi SSID-s and other Wi-Fi parameters upon detection, acceleration values along the three axes and GPS coordinates.

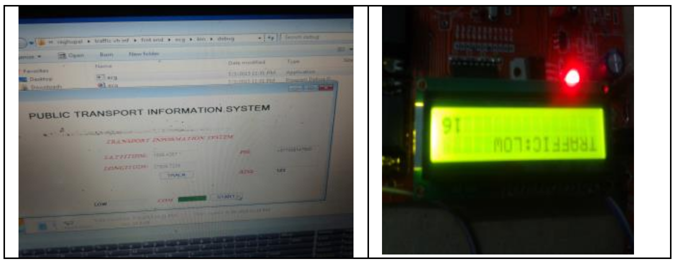

Fig : traffic at low

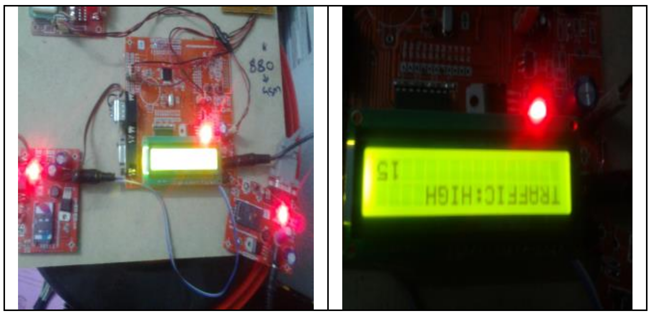

Fig : traffic at High

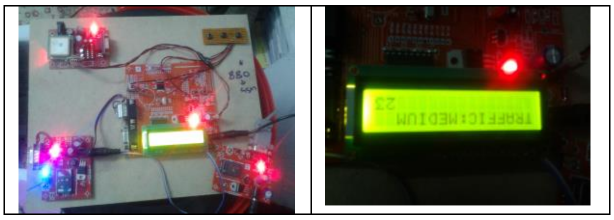

Fig : traffic at medium

We store the collected sensor data in CSV (Comma Separated Values) format. Note, that constantly monitoring these sensors is energy intensive and straining the phone's battery limits the device up-time that would discourage people from using TrafficInfo. We collect these sensor measurements only for experimental purposes to process them later, in an off-line manner. At the moment, we have been analyzing these traces to identify what sensor readings with which sampling intervals are needed to detect the different event types, and we will implement the necessary client and server side logic accordingly.

The measurement module provides also a user interface (see Fig. 6) for manually indicating rapid acceleration, halt events at stops, and whether the user is going on foot or traveling by vehicle. For some pre-selected transportation lines we hardcoded stop information to speed up the logging process and to minimize the human error factor (unintentionally giving misinformation concerning **stop names).**

E. Advanced Features

We have implemented some additional features beyond the core functionality to make the use of TrafficInfo more conve-nient. Thus, the user has an option for layer selection to avoid an overcrowded layout. Here the user can mark those traffic lines from the subscribed channels that are of interest since hiding the others. The crowdedness information is depicted.

SUMMARY

In this demo paper, we presented a participatory sensing based real-time public transport information service and its front-end, Android based smart phone application called TrafficInfo. It is aimed to monitor and provide real-time traffic updates to the passengers about the public transportation lines of any modern city in case of sufficient participatory users.

TrafficInfo visualizes live traffic updates on the Google Map of the city, offers reporting options to the users and collects measurement data from the phone's built-in sensors.

Our future plans include: automate the data gathering process and event detection; and provide the user with even more advanced features, putting the emphasis on user comfort. We believe that limiting the required user interaction only to voluntary reporting on unusual incidents can increase substantially the user experience. Moreover, we plan to open up our system towards other applications which are able to gather the required context sensing data, because data collection and live public transport visualization can be pretty much detached from each other.

REFERENCES

[1] R. Ganti, F. Ye, and H. Lei, "Mobile Crowdsensing: Current State and Future Challenges," *IEEE Communications Magazine*, pp. 32–39, Nov. 2011.

[2] P. Saint-Andre, "Extensible Messaging and Presence Protocol (XMPP): Core," RFC 6120 (Proposed Standard), Internet Engineering Task Force, Mar. 2011. [Online]. Available: http://www.ietf.org/rfc/rfc6120.txt

[3] R. L. Szabo and K. Farkas, "A Publish-Subscribe Scheme Based Open Architecture for Crowd-sourcing," in *Lecture Notes in Computer Science 8115: Proceedings of 19th EUNICE Workshop on Advances in Communication Networking (EUNICE 2013)*. Springer, Aug. 2013, pp. 287–291.

[4] Google Inc., "General Transit Feed Specification Reference." [Online]. Available: https://developers.google.com/transit/gtfs/reference