Towards the Utilization of Crowdsourcing in Traffic Condition Reporting

P. Rantanen*, P. Sillberg * and J. Soini*
* Tampere University of Technology, Pori, Finland
petri.rantanen@tut.fi

Abstract - The use of traffic information in map applications designed for stand-alone navigation devices as well as in mobile devices has become a common trend. This information is often governed by the various service providers with little or non-existent feedback from the users. Using a wide user base it is possible to collect information on traffic conditions faster and more efficiently. Additionally, many of the events faced on the road can be challenging to detect by automatic means, but are easily noticed by the road users – animals on the road and broken or missing road signs are only a few examples.

To better facilitate the utilization of information gathered from road users, simple and easy-to-use software solutions are required. This paper presents a prototype mobile application, which the road users can take advantage of for both following the on-going traffic conditions while driving and for collectively reporting traffic events other users might be interested in. The high-level architecture, application and data utilized in the reports are presented in addition to the preliminary findings of the on-going research. This paper will also discuss the challenges identified while developing the application.

I. Introduction

There have been several studies on how to take advantage of automated means in the traffic environment. Topics such as the detection of pavement conditions [1], winter conditions [2], vehicles [3][4] and road signs [5] have been of constant interest for the past decade. There is little doubt that the use of automated means will increase in the future. Nevertheless, events such as animals on the road, broken road signs and poor visibility are easily recognized by humans, but can be challenging use cases for automatic methods. Thus, the potential wide user base offered by the people travelling on the roads should be more tightly integrated into the road condition reporting.

Most map applications provide "layers" that show basic details (for example, road works, traffic jams) concerning the current road conditions, but generally, no option is given to the user to give feedback on the accuracy of the information. In Finland, there is an official telephone number provided by the local road agency the road users can call to report bad road conditions, and several radio stations accept calls concerning events met on the road. Unfortunately, these services are very seldom connected to the devices and the services the people use in their everyday lives. In essence, the application (or service) presented in this paper could be seen as a more modern replacement for these telephone-based services currently available.

The initiative to develop the application presented in this paper came from one of the corporate partners (a logistics and transportation company) of our project. They had a need for a very simple and easy-to-use application that the professional drivers of the company could use to notify each other about various road incidents, events and road conditions that could affect the deliveries.

This paper presents the preliminary results of the APILTA (Avoin PIlvipalvelukonsepti joukkoistetun Liikennedatan TArpeisiin, or "Open Cloud Platform for Crowdsourcing Based Services", in English) project currently on-going in Tampere University of Technology, Pori Department. The primary purpose of the research within the context of this paper is to design an easy to use application (user interface) that would not distract the driver and the users would be willing to take advantage of while travelling. Keeping this goal in mind, the following section (Section II) discusses the related studies, Section III provides a high-level look on the architecture and the use case in general, Section IV will describe the application itself, Section V offers a discussion on the challenges and opportunities, and finally, Section VI summarizes this paper.

II. Related Studies

The use of traffic data in crowdsourcing applications is not a completely new idea. For example, in [6] the focus was on evaluation of anonymized data created by an electronic travel journal to be used as study material for mobility research. If the collected data was accurate enough, it could be used in design of the traffic network. There are efforts to estimate the traffic conditions with data that are gathered from different sources and by different methods (e.g. traffic monitoring by inductive loop detectors, satellite positioning, and mobile phone positioning) [7][8]. To reliably evaluate the traffic conditions a lot of traffic monitoring is required. This is where the crowdsourcing can become a feasible source of data.

Efforts for the concept of Smart City (SC) can be useful as transportation and traffic are typically studied together with this topic. Sensor networks, data collection and data management are key components in development of Smart City applications. Literature review [9] classifies a wide range of architectures and technologies used in the field of SC. The newest trend seen was the introduction of IoT (Internet of Things) and in the future IoE (Internet of Everything) is expected to be the next important milestone. IoT enabled SC applications often employ Smart Objects (SOs) in the monitoring of traffic and environment. A large number of devices, software and services (e.g. mobile phones or even vehicles) can be categorized as a SO [10] [11].

In the use case of Smart Transportation [10], the sensor data is seen to have a crucial role in managing the transport system. The trustworthiness, robustness, timeliness, and security of the communicated data together with maintaining the privacy of users is of major importance in a SC system [10]. The security and reliability of a cloud-based city management platform have also been studied [12].

There have been studies on using mobile phones as location location sensors in a Smart City service for car sharing to improve the traffic conditions in the City of Barcelona [11]. Furthermore, there has been research on utilizing event-based architecture to conduct Machine to Machine (M2M) communication in vehicular context [13], the Internet of Vehicles (IoV) and the Social Internet of Vehicle (SIoV) [14], detecting road conditions using smartphone sensors [15], and analyzing video with a computer vision system to recognize overall road conditions and the status of traffic infrastructure [16].

There are several applications that are similar to our approach. Beat the Traffic [17] used to offer real-time traffic information partially provided by users in the United States and Canada. Waze [18] similarly offers its users to participate in gathering new traffic information. Inrix [19] provides a more complex service for traffic management including several mobile applications for tracking traffic conditions, but lesser options for user to manually create reports. In Finland, a new pilot (NordicWay Coop) was launched last year in co-operation with industry partners and the Finnish Transport Safety Agency with a goal of allowing people to report traffic conditions [20].

When it comes to reporting new alerts, each application has a slightly different approach: Beat the Traffic offered more complicated forms and a simpler "shake to report" feature; Waze uses speech recognition; and NordicWay Coop utilizes a more traditional menu-based approach. Based on the literature each controlling method (voice recognition [21][22][23], touch, tactile and gesture [24]) have their own advantages and disadvantages with perhaps no clear winner to be found amongst the options.

Similarly to Beat the Traffic, Inrix and NordicWay Coop, our application utilizes touch-based controls with the main difference being in the user interface design – the look'n feel of the application and the user interface "flow" when creating a new alert. In our case, the goal is on enabling an easy reporting of alerts where as many of the commercial applications are primarily focused on map-based features (e.g. navigation).

As mentioned before, the initiative for our application development came from a logistics and transportation company. An individual company might have the need to report alerts that are not by default included in the system, and the company might not be willing to share the reports outside the company, or the company might only trust the alerts received from a particular user group. User groups and customization would also allow regular users to team up with friends or colleagues for sharing alerts in a more social way. With the exception limited features offered by the Waze service there seems to be apparent lack of customization and user grouping options in the existing applications.

III. Architecture

The system architecture (seen in Fig. 1) is based on receiving alerts from different clients, and transmitting the alerts to clients who are or would be affected by them. The service is implemented by utilizing commonly used open source components such as Apache Tomcat, Apache Solr and Oracle MySQL.

In the first prototype of the client, the available alert types are predefined to contain just a few examples, but the architecture and alert syntax itself are extendable. Also, the system can accept photos attached to the alerts, but this feature is not used in the application presented in this paper. City road maintenance is one use case where the additional info could be useful.

The alerts on the client are updated by regularly polling the representational state transfer (REST) Application Programming Interface (API) to retrieve the newest alerts that affect the user. Thus, the clients do not need to subscribe to listen for new alerts. The client would need to continuously report its location to the service to receive up-to-date alerts in any case, but in our approach the service will immediately respond with alerts that are in range. This would also allow anonymous or unregistered users to use the API. In our case, because of the limited computing resources available, the service is open only for users accepted to participate in our piloting phase. In practice, there could also be real business cases (e.g. pay-to-use, providing user preferences across devices based on the user account) that require the users to be authenticated before using the service.

Each alert reported by the clients includes the geographical coordinate of the event and the search method provided by the API accepts a polygon based area filter and a heading based filter. This way the client device can construct more accurate search terms that cover more area in front of the device than behind. The client can also adapt to the increasing speed by retrieving the alerts more often, or by growing the search area bigger.

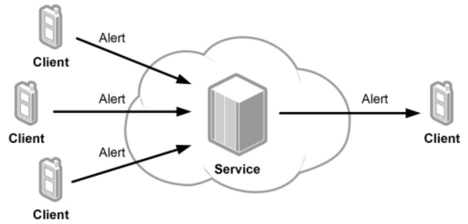

Figure 1. High level architecture.

In the first phase of the pilot tests, there are only a few alert types to be included (animals, broken street signs, slow traffic). The architecture is built to accept new, user created alert types for extending the usability of the system. Any authenticated user can create a new alert type, but by default the application presented in this paper will only utilize the built-in types. New types must be configured (chosen from a list of types known by the service) in the application settings – this is to prevent users from spamming the service with unnecessary global alerts, but the approach will still allow the utilization of custom tags with relative ease of use.

The API provides the data in Extensible Markup Language (XML) documents by default, and supports JavaScript Object Notation (JSON) format as well. The data format of the alert includes a timestamp, location (coordinate), alert type, alert description and information about how long the alert is valid. The validity period of the alert varies case by case, and at the moment there is no good approximation when or how a certain alert should become inactive. In accidents and similar events an official user such as the police or ambulance staff could be given the permission to remove the corresponding alert, but in the case of animal alerts the validity period can be challenging to predict. The default validity period for each alert can be defined when the alert type is created in the service. In our use cases, alerts will be automatically invalidated within a couple of hours. In principle, the validity period could also be set by the client though choosing the correct time value when posting a new alert would probably be just as difficult as choosing a correct pre-defined validity time.

IV. Prototype Application

As the main use case is driving a car, this poses certain requirements for the application. Most importantly, the application should not distract the driver. For this goal, the application has been designed to require as little interaction as possible with emphasis on automation both in the information gathering and in the flow of the user interface.

There are no technical limitations why the application could not be used, for example, when walking, and alternative means of transport could provide possibilities for features purposefully not implemented in the current version. Features such as alert rating (feedback) or reporting of incorrect alerts, or zooming and panning of the map view could be easily managed by the user when walking, but could be too distracting when driving a car.

The application has two basic functions (or features): reporting new alerts to the service; and showing alerts that are near to the user. The features are shown in Fig. 2 and in Fig. 3. Additionally, the application contains a settings view (not visible in the figures), which allows the user to change the common configuration parameters such as the credentials used to authenticate the user with the service. The settings view can also be used to select the alert types the user wishes to listen for new alerts and the option to select the types available for reporting new alerts.

The screenshots are taken from a Google Nexus 7 device running the Android 7 operating system. The application should work on any relatively recent (even low-end) Android phone or tablet with a touch screen and an Internet connection. As the transferred data consists of simple Hypertext Transfer Protocol (HTTP) GET and POST requests with small XML or JSON payloads (depending on the amount of alerts, generally from ten to 50 kilobytes) even a relatively low bandwidth network connection can be utilized.

A. *Feature 1: Report a New Alert*

Fig. 2 illustrates the user interface flow applied when the user reports a new alert. The first picture (left side, Fig. 2) shows the default (idle) view visible when no alerts are in range. In this example case the user has configured the application so that there are two alert types available for reporting new alerts: reports for animals on road (center, Fig. 2); and reports for slow moving traffic (right, Fig. 2).

To browse the available alert types the user only needs to touch anywhere on the screen. This will automatically show the next view: from the default view to the "animal on road" view; from the "animal on road" view to the "slow traffic" view; and from the "slow traffic" back to the default view – as seen in the Fig. 2. A move from one view to another is accompanied by a sound (the spoken name of the alert type). This allows the use of custom alert type names by utilizing text-to-speech features provided by the Android platform.

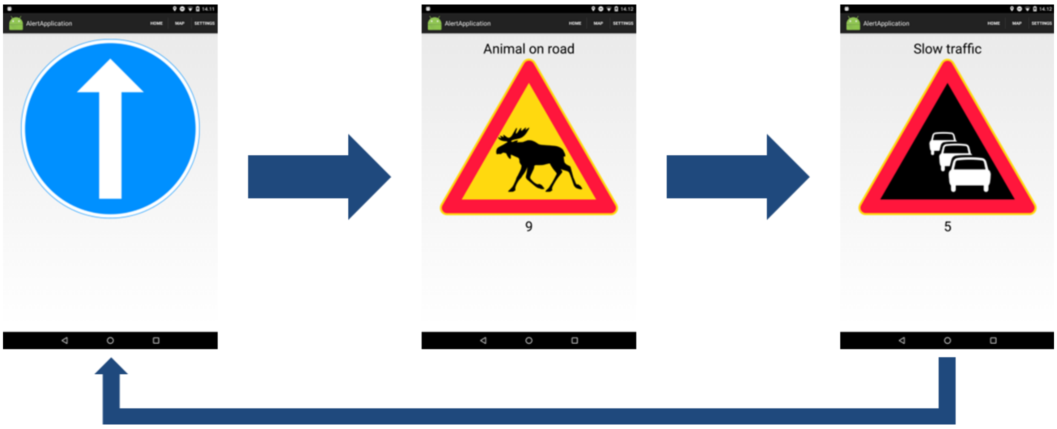

Figure 2. User interface flow when user reports a new alert.

If the user does not touch the screen within a predefined time and the active screen is one of the alert views, a new alert (with applicable details such as timestamp and user's location) will be automatically sent to the service. The time left until the alert will be sent is indicated to the user by a countdown timer visible on-screen (as can be seen in Fig. 2 middle and right, below the icons).

The time until the alert is posted can be configured from the application and a longer time will naturally give the user more time react. Touching anywhere on the screen while the counter is running will abort the operation and screen will change to show the next view. If the next view is an alert view a new counter will start, if the next view is the default view the application will return to the idle state.

The audio cues and the fact that the user can touch anywhere on the screen are designed to allow navigation without looking at the screen. On devices that do not have a touch screen the navigation could also be implemented by providing a single button to use for changing the views. The user interface is also meant to be simple enough to be usable on various screen sizes. Alternative approaches such as multiple buttons provided for navigation or a scrollable list of alerts could cause difficulties for the driver for using the application especially on devices with smaller screen sizes.

In principle, there could be any number of alert types to choose from, and if more alert types were used, new views would appear in the user interface cycle (for example, in Fig. 2, a view with a new traffic sign could be between the elk sign and the slow traffic sign). In practice the user should use the application settings to configure only the alert types he/she is planning to use to be visible in the user interface to reduce the total amount of browsing needed.

Another option could be to allow the user to speak the name of the alert type when choosing the alert to be reported, but in a location with a potential ambient noise (e.g. in a car) it can be challenging to implement reliable speech recognition. Additionally, this would require the user to remember the actual names of the alerts or implementation should be smart enough to decipher the user's meaning from alert names pronounced only partially correct. Furthermore, as the application is entirely separate from the in-car systems, the voice control could also interfere with the built-in speech recognition features of the car itself or the use of two separate systems could confuse or annoy the user.

B. Feature 2: Show Nearby Alerts to the User

The user interface flow for showing the nearby locations is can be seen in Fig. 3. As long as the visible view is the default (idle) view (Fig. 3, left) the application will automatically show the map when alerts are detected within a pre-configured range or the travel time to the alert is within a pre-defined threshold (for example, the location indicated by the alert is within 100 meters, or the location would be reached in 30 seconds). The change to the map view is also indicated by an audio cue. If the user is currently browsing for alert types and is not in the default view, only a sound cue is played and the application will wait for the user to return to the default view before showing the map view. After the alerts go out of range, the application will automatically switch back to the default view. The user can also change to the map view by tapping the "Map" option on the top right, and the user can return from the map view to the default view (and to reporting new alerts) by simply touching anywhere on the map screen or by touching the "Home" option on top right. The "Home" option can also be used to return to the main view from any other view, and can be used at any time to cancel a reporting of a new alert.

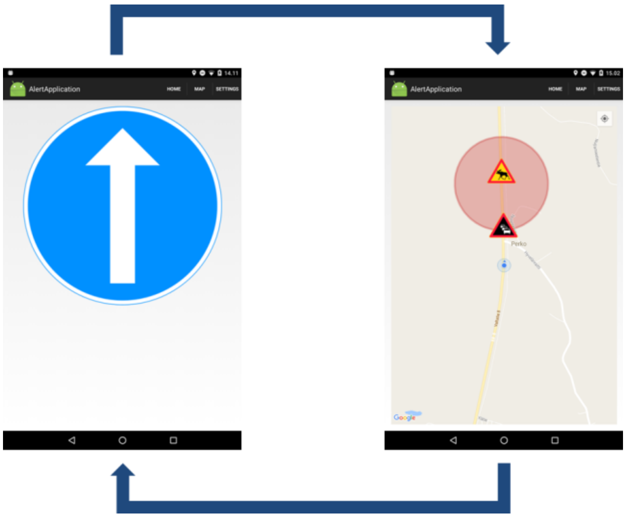

Figure 3. User interface flow when an alert becomes in range.

In the current version of the application, the map will automatically zoom to and follow the user's current location. Manual panning and zooming of the map often implemented by touch gestures are disabled. The primary purpose of this is to allow quick and easy navigation back from the map view.

The upcoming alerts are illustrated using the same icons as seen in Fig. 2. In our use case, the alerts are represented by road signs, which are easily recognized and associated by the driver to the events in question. Naturally, in many cases new icon designs are required as not all alert types can be illustrated by road signs alone. If the range of the event is known, it is illustrated by a transparent area around the icon. In Fig. 2, the "Animal on road" event contains range information, but the "Slow traffic" event does not. In the current service implementation, the area is visible only on alert types that have the range predefined (e.g. animal warning have a default range of 200 meters). The prototype application (user interface) does not support the creation of new alerts with range defined.

In the background the application will continuously poll the service for every couple of minutes requesting new alerts (i.e. alerts reported after the previous request) within a larger area (for example, within two kilometer radius) around the user's location. The retrieved alerts are cached by the application. The cache is checked whenever the user's actual location changes (within certain accuracy) and alerts that have moved too far from the user are automatically removed from the cache and alerts that are close enough are shown in the map view.

The new alerts could be requested from the service each time the user's location changes as opposed to more relaxed background polling, but this would both increase the amount of requests on the server-side as well as use more power and network bandwidth on the device itself. The downside is that the alerts do not appear on the client devices in real time, but in practice, real-time updates are seldom needed.

V. DISCUSSION

In essence, there have been two main challenges in developing the application: to make the user interface as easy to use as possible; and to validate the achieved results in the larger context of crowdsourcing.

To further aid the use of the application, automation would provide assistance for the user, and there are several promising, emerging and already implemented technologies such as speech recognition, image analysis (for example, the detection of objects or events from frames captured from a video camera), and the utilization of various sensors embedded in road modern road vehicles. Unfortunately, the most in-car systems cannot be directly accessed or extended with new features either because of security reasons or closed nature of the systems. This also means that buttons located, for example, in the steering wheel cannot be easily used to control a smart phone application.

Similarly, extending commonly used navigator devices and map applications can be a challenging task. Several companies (e.g. Google, Microsoft) provide online APIs for navigation applications, but the utilization of these APIs often require redundant work for the implementation of basic navigation features already present in the mobile device itself. Due to the fact that the user would need to use multiple applications for two closely related tasks (for example, the application presented in this paper and a proper navigation application) may decrease the user's willingness to use the provided, otherwise advantageous features. In an optimal situation, the functionalities presented in this paper would be integrated with an existing system or device, preferably in a navigation application.

As explained earlier, this paper presents only the preliminary results of an on-going study. The piloting phase utilizes people participating in the project either directly in the university or in the co-operating companies. The initial feedback on the idea has been positive, but because of the smaller user base, it is difficult to validate whether a larger audience would be willing to participate in reporting the alerts. A wider user base would be required for conclusive results, but unfortunately organizing a larger crowdsourcing effort can be challenging both for finding the applicable audience and because of the increased (computing) resources required for hosting the service online. In principle, it could also be possible to realize a working service even with a help of a smaller, but more motivated group enthusiasts.

Another issue, closely related to the widening of the target audience is the validation of the alert reports the users have submitted. Within a smaller and more controlled user base it is easier to protect the service against malicious use. By nature the reports are only "alerts" and, for example, spamming the service with incorrect alerts should not pose danger to the users (drivers), but it might render the service unusable, or at the very least, lower the users' interest in the service. Possible solutions could be to allow "rating" of the alerts or to require a certain amount of alerts of the same type to be reported within an area before the alert is shown publicly.

In the first solution, the primary problem would be how to implement the feature in the application so that it does not distract the driver. On one hand, it is entirely possible that the feature would take the user's attention away from driving causing more accidents than the application would prevent. On the other hand, if the feature would be implemented on a separate web portal, it is unknown how many of the users would bother to report the alerts after arriving at their destination.

The problem with the seconds approach is that if there are only a few users that report alerts using a road, the reported alert may never be published, or it might take a longer time before the reported alert becomes public causing potentially a larger number of road users to miss the alert. As the service does not allow anonymous access, rating alerts and users could (at least in theory) in time filter out users that post unnecessary or incorrect alerts. In the case of animals on the road, one way to verify a low number of reports could be done by making a cross-check with another database with details of animal density in that area to determine the chance for a such event to occur.

Information and guidelines for minimizing driver distraction have been released by [25][26]. Furthermore, there have been studies on user interface and system response design in the context of minimizing driver distraction [27]. However, the guidelines are only applicable to original equipment electronic devices, and do not cover aftermarket or portable devices. This means that because there is less control over mobile applications in official requirements, the applications can be organized in a more liberal way, possibly even in a distracting manner to the driver. Thus, it can be easy to overlook traffic safety aspects, especially if the recommendations would interfere with the designer's vision for the look'n feel of the application.

In conclusion, it can be said that there remain several currently unresolved and challenging tasks. Regardless, the authors of this paper believe that more could be achieved for the goals of averting accidents and reporting road conditions through better integration of road users' feedback potential into the overall road ecosystem.

VI. SUMMARY

Despite the progress in the development of automatic means to detecting road conditions, there are still cases where humans are more capable of detecting the events faced while travelling. Yet, the commonly used navigation devices and applications seldom offer any means for the user to provide feedback on the accuracy of the road information.

To better realize the potential of the road users in reporting road conditions, easy to use applications that the users can take advantage are required. This paper presented an example of an application the road users can utilize to report events faced while travelling by car. The high-level architecture utilized in the overall service was briefly introduced in combination of the data used to describe the alerts. Additionally, this paper provided a discussion on the challenges and unresolved issues identified while designing the application.

References

[1] S. C. Radopoulou and I. Brilakis, "Improving road asset condition monitoring," Transportation Research Procedia, vol. 14, pp. 3004-3012, June 2016.

[2] R. Omer and L. Fu, "An Automatic Image Recognition System for Winter Road Surface Condition Classification," 2010 13th International IEEE Conference on Intelligent Transportation Systems (ITSC), Madeira Island, Portugal, pp. 1375-1379, September 2010.

[3] S. Sivaraman and M. M. Trivedi, "Active learning for on-road vehicle detection: a comparative study," Machine Vision and Applications, Vol. 25, Issue 3, pp. 599-611, April 2014.

[4] S. Sivaraman and M. M. Trivedi, "Looking at Vehicles on the Road: A Survey of Vision-Based Vehicle Detection, Tracking, and Behavior Analysis," IEEE Transactions on Intelligent Transportation Systems, Vol. 14, Issue 4, pp. 1773-1795, December 2013.

[5] Y.-Y. Nguwi and A.Z. Kouzani, "A Study on Automatic Recognition of Road Signs," IEEE Conference on Cybernetics and Intelligent Systems, Bangkok, Thailand, pp. 1-6, June 2006.

[6] H. Keskikiikonen, "The possibilities of crowdsourcing in travel surveys," Master's Thesis, 2014.

[7] S. Kytö, "Quantifying car traffic fluency with geospatial methods: Case Helsinki metropolitan area," Master's Thesis, 2016.

[8] H. Mattila, "Real-time estimation of traffic conditions on road sections [Linkkikohtaisen liikennetilanteen ajantasainen arviointi]," Helsinki 2003. Finnish Road Administration. Finnra Reports 61/2003. 175 p. + app. 13 p. ISSN 1459-1553, ISBN 951-803-179-7, TIEH 3200848-v.

[9] C. Kyriazopoulou, "Smart city technologies and architectures: A literature review," 2015 International Conference on Smart Cities and Green ICT Systems (SMARTGREENS), pp. 1-12, May 2015.

[10] E. Z. Tragos et al., "Enabling reliable and secure IoT-based smart city applications," 2014 IEEE International Conference on Pervasive Computing and Communication Workshops (PERCOM WORKSHOPS), Budapest, Hungary, pp. 111-116, 2014.

[11] C. Doukas and F. Antonelli, "A full end-to-end platform as a service for smart city applications," 2014 IEEE 10th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), Larnaca, Greece, pp. 181-186, 2014.

[12] W. Li, J. Chao and Z. Ping, "Security Structure Study of City Management Platform Based on Cloud Computing under the Conception of Smart City," 2012 Fourth International Conference on Multimedia Information Networking and Security, Nanjing, China, pp. 91-94, 2012.

[13] J. Wan, D. Li, C. Zou and K. Zhou, "M2M Communications for Smart City: An Event-Based Architecture," 2012 IEEE 12th International Conference on Computer and Information Technology, Chengdu, China, pp. 895-900, 2012.

[14] K. M. Alam, M. Saini and A. E. Saddik, "Toward Social Internet of Vehicles: Concept, Architecture, and Applications," in IEEE Access, vol. 3, pp. 343-357, 2015.

[15] A. Ghose, P. Biswas, C. Bhaumik, M. Sharma, A. Pal and A. Jha, "Road condition monitoring and alert application: Using in-vehicle Smartphone as Internet-connected sensor," 2012 IEEE International Conference on Pervasive Computing and Communications Workshops, Lugano, Switzerland, pp. 489-491, 2012.

[16] Vionice Ltd, "RoadAI - Record, View, Share & Benefit," 2016. Retrieved February 6, 2017 from https://vionice.fi/en/service/2

[17] Wikipedia, "Beat the Traffic," 2017. Retrieved March 31, 2017 from https://en.wikipedia.org/wiki/Beat_the_Traffic

[18] Wace Mobile, waze, 2017. Retrieved March 31, 2017 from https://www.waze.com

[19] Inrix, 2017. Retrieved March 31, 2017 from http://inrix.com

[20] HERE, "Public pilot of HERE-powered NordicWay Coop launched in Finland". Retrieved March 31, 2017 from http://360.here.com/2016/06/02/public-pilot-of-here-powered-nordicway-coop-launched-in-finland

[21] A. Barón and P. A. Green, "Safety and usability of speech interfaces for in-vehicle tasks while driving: a brief literature review," Technical Report UMTRI-2006-5, University of Michigan Transportation Research Institute, Ann Arbor, Michigan, USA, 2006.

[22] Z. Hua and W. L. Ng, "Speech recognition interface design for in-vehicle system", Second International Conference on Automotice User Interfaces and Interactive Vehicular Applications (AutomotiveUI 2010), Pittsburgh, Pennsylvania, USA, pp. 29-33, 2010.

[23] J. C. Chang, A. Lien, B. Lathrop and H. Hees, "Usability evaluation of a Volkswagen Group in-vehicle speech system," 1st International Congerence on Automotive User Interfaces and Interactive Vehicular Applications (AutomotiveUI '09), Essen, Germany, pp. 137-144, 2009.

[24] K. M. Bach, M. G. Jæger, M. B. Skov, N. G. Thomassen, "You can touch, but you can't look: interacting with in-vehicle systems," SIGCHI Conference on Human Factors in Computing Systems (CHI '08), Florence, Italy, pp. 1139-1148, 2008.

[25] Department of Transportation, "Visual-Manual NHTSA Driver Distraction Guidelines For In-Vehicle Electronic Devices," 2012.

[26] European Transport Safery Council, "'PRAISE': Minimising In-Vehicle Distraction," 2010.

[27] J. Sonnenberg, "Service and user interface transfer from nomadic devices to car infotainment systems," In Proceedings of the 2nd International Conference on Automotive User Interfaces and Interactive Vehicular Applications (AutomotiveUI '10). ACM, New York, New York, USA, pp. 162-165, 2010.