Article

A New Location-Based Services Framework for Connected Vehicles Based on the Publish-Subscribe Communication Paradigm

Kachane Sonklin [1,†,‡], Yanming Feng [1,*], Dhammika Jayalath [1,‡], Charles Wang [1,‡]

[1] Science and Engineer Faculty, Queensland University of Technology, Brisbane QLD 4000, Australia;kachane.sonklin@hdr.qut.edu.au; y.feng,dhammika.jayalath,cc.wang@qut.edu.au
[*] Correspondence: y.feng@qut.edu.au; Tel.: +61-7-31381926
[‡] These authors contributed equally to this work.

Abstract: Location-Based Services (LBS) have been widely deployed for the connected vehicle (CV) applications such as vehicle navigation,vehicle tracking and location-based augmented reality. The current LBS deployments have limitations in supporting time-critical CV use cases, including vehicle to vehicle (V2V), vehicle to infrastructure (V2I) and vehicle-to-people (V2P) safety applications. The paper presents the new LBS framework based on the publish-subscribe communication paradigm, to enable device-to-device (D2D) connections through use of selected application protocols in the application layer of the TCP/IP layered protocol model. Two publish-subscribe application protocols, Distributed Data Service (DDS) real-time publish and subscribe (DDS-RTPS) and Message Queue Telemetry Transport (MQTT), are introduced to support the LBS D2D applications. A number of test scenarios with Mosquitto MQTT and OpenDDS under 4G-mobile broadband (MBB) services are designed to assess the transmit/receive round-trip time (RTT) and packet-loss rate (PLR) with settings of a publisher to multiple subscribers, to simulate the connections to multiple vehicles. The transmission frequency is set for 10 Hz and the message sizes vary from 100 to 2000 Bytes. The PLRs are defined as the percentages of the delayed messages beyond a delay limit. Static test results with OpenDDS show that for the RTT delay beyond the limit of 100 ms, the total PLRs range between 5.25% and 8.76% for the message size of 50 to 2000 Bytes. Vehicle testing results with Mosquitto show that PLRs for the RTT delays between 200 ms and 1000 ms are 0.63%, 3.58% and 5.77%, for connections with 1, 4 and 10 vehicles, respectively. The results demonstrate the potential of the D2D LBS framework for medium-demanding CV safety applications such as V2P and V2I use cases, taking advantages of the 4G-MBB services and 5G extreme mobile broadband (eMBB) services and mobile devices generally available with all road users.

Keywords: Location-Based Services,Vehicle-to-Everything(V2X), Publish-subscribe, Application Protocol

---

1. INTRODUCTION

Location-based services (LBS) is traditionally defined as an information service, accessible with mobile devices through mobile network and utilizing the ability to make use of the geographical position of the mobile device [1]. Today, users' ability to generate content is core to the services or applications. A more recent definition of an LBS is a service where [2]

- The user can determine their location.
- The information provided is spatially related to the user's location.
- The user is offered dynamic or two-way interaction with the location information or content.

In this way the user can answer three key questions: Where am I? What can I do nearby? What is my assessment of about this place? Referring to this definition, there have been many well-defined LBS applications, including vehicle and personal navigation, location-based social media, mobile

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 January 2019 doi:10.20944/preprints201901.0193.v1

Location-Based gaming, personal and vehicle tracking, monitoring, emergency, marketing, information services and augmented reality [3]. vehicle-related location-aware applications are summarized below.

- **Road navigation:** With this service, user's mobile device can search for a point of interest and travel on the best travel routes between the user's current location and end location. A Google mobile map is one of the most popular navigation applications which combine the road maps and hybrid positioning technologies to give a user high position availability and timely real-time turn to turn instruction services.
- **Vehicles tracking:** A LBS server for vehicle tracking accesses the locations of individual vehicles through Internet connections with onboard vehicle tracking devices. Vehicle related information and locations can be displayed on the maps and accessible via user interface apps. For instance, a Uber taxi user can view a number of Uber cars nearby. Vehicle tracking systems are more commonly used by fleet operators for fleet management functions such as fleet tracking, routing, dispatching, on-board information and security.
- **Monitoring:** A vehicle monitoring system is one of the LBS services that combine some technologies, such as Global Navigation Satellite System (GNSS), onboard vehicle sensors to follow something based on geographical position and time. While vehicle tracking is concerned with the historical and current data of the vehicles, location monitoring has more to do with knowledge such as exceptions, alerts, and warnings based on a set of rules, normally bound by space and time [4].
- **Emergency services:** Emergency service is one of the most important LBS applications that can identify and locate an individual who is calling or signalling to the emergency services, such as ambulance and police because of an emergency situation. This location may be a physical address or coordinates of a moving vehicle. The caller's telephone number is used in numerous manners to track a location that can be used to dispatch police, fire, emergency medical and other response resources [5].
- **Augmented Reality:** Augmented Reality (AR) is a combination of the virtual world with real environment by overlaying digital information including images, sound, or geographic location, on the top of the existing environment, such as streets and buildings. It provides real-time interaction and registered in 3D to users through screen of mobile devices onboard vehicles.

These LBS applications are typically implemented with mobile devices carried by people or installed onboard vehicles through Internet connections. One common characteristic is that the most communications follow the client-server architecture. It employs point-to-point data link and two-way interactions. For instance, the application-layer protocol, HyperText Transfer Protocol (HTTP), used primarily on the World Wide Web, uses a client-server mode. The web browser is the client and communicates with the web server that hosts the website. To establish the connection between the client and the server, the client begins to send a request to the server, and then the client waits for the server response. The server sends the response back by re-establishing the connection with the client side. In general, the current LBS, typically described as Device-to-Server (D2S) or Server-to-Device (S2D) LBS, is a centralised system. In fact, the update rates of mobile location data are usually set low, such as 5 to 30 seconds in vehicle tracking or fleet management. Moreover, most of the computation is completed on the LBS server side, such as real-time vehicle tracking, alternative route suggestions and emergency information. The LBS server takes normally seconds to minutes for computation and analysis. In the context of connected vehicles, the location-based D2S and S2D communications are referred to Vehicle to Network (infrastructure), or V2N/V2I connections. However, connected vehicles (CV) more generally refer to the presence of devices in an automobile that connect devices within the car/vehicles together or with devices, networks and services outside the car including other cars, people, home, office or infrastructure. In the V2I communication, real-time data, such as road traffic and weather information can be sent and received to/from the road side unit infrastructures through the core networks, while the V2V communication allows the vehicles communicate each other [6]. More specifically connected vehicles include the concepts of Vehicle-to-Everything (known as V2X) interactions, such as Vehicle-to-Vehicle (V2V), Vehicle-to-People (V2P) and Vehicle-to-Network (infrastructure) data or message exchanges. In the V2X scenarios, it requires device-to-device connections supporting one-to-many and many-to-many communications. The mobile devices/platforms often provide data streams and images to the servers or other devices instead of information or context-aware messages. The computation or decision making should be completed at the mobile device edge in real-time instead of at a server with a delay. V2X communications requires high timeliness, low round-trip time (RTT) latency, and high user scalability. In support of wireless connectivity among vehicle-based devices, and between fixed roadside devices and vehicle-based devices, the currently deploy-able technologies define the hardware and services operating from the application layer down to the physical layer. These include the 5.9 GHz Dedicated Short-Range Communications (DSRC)/IEEE Standard for Wireless Access in Vehicular Environments (WAVE), and 4G-Long-Term Evolution (LTE). The developing technologies include LTE-Direct and LTE-Vehicle and 5th generation cellular technology (5G) Ultra-reliable Machine-Type communications (uMTC) services, which are also hardware dependent. On the other hand, the development of the Internet of Things (IoT) applications drives the merge of DSRC, wireless local area networks (WLAN), and cellular networks, leading to the Internet of Vehicles (IoV). IoV refers to the real time interactions between vehicles, and vehicles with other road entities [7]. IoV is a large-scale complex system with heterogeneous network components and diverse applications, which are enabled by special design of MAC layer protocols and routing protocols and application layer software tools [8],[9]. The Internet-based LBS applications are implemented only at the application-layer, thus having the potential advantages for implementation of V2X applications without depending on physical-layer changes. In addition to vehicle-based navigation and tracking applications, there have been numerous well-adopted consumer grade LBS applications in people's daily life, thanks to the explosions of mobile devices and social networks and the global leading LBS providers such as Alibaba, Apple, Foursquare, Google, Uber and HERE. The high popularity of the LBS can leverage the rapid deployments of more critical V2X LBS applications for road safety, efficiency and mobility benefits. The problem is that to support the time critical V2X applications, the limitation of the current LBS client-sever architecture must be addressed and the new communication architecture should be adopted. To what degree the new LBS architecture can meet the timeliness and reliability requirements of V2X applications is the next key question to answer.

In the recent work[10], a new LBS definition was proposed for time critical applications and LBSs were extended from the consumer and business categories to the professional category. In this work, the contributions are twofold: (1) the establishment of the D2D LBS framework and (2) extensive experimental assessments to demonstrate the feasibility and potential of the D2D framework in supporting light-weight V2X applications. The rest of the paper is arranged as follows. First, we review the current LBS communications based on the Request/Response framework. Next we present a D2D LBS framework based on the publish-subscribe architecture. The publish-subscribe paradigm offers advantages such as lower latency and higher scalability, which can support D2D LBS applications. After this, we examine two application protocols that can support the publish-subscribe paradigm, including Distributed Data Service (DDS) real-time publish and subscribe (DDS-RTPS) and Message Queue Telemetry Transport (MQTT). We design DDS and MQTT test scenarios to examine the transmit/receive round-trip-time (RTT) latency and packet-loss rates (PLR) with different latency bounds. Experimental results are presented to show the RTT and PLR performance of D2D data exchanges through 4G-mobile broadband (MBB) services at the frequency of 1 to 10 Hz and different message sizes, demonstrating the benefits of the new LBS framework for V2X LBS applications.

2. RELATED WORKs ON LBS ARCHITECTURE AND COMPONENTS

A LBS architecture mainly refers to the communication paradigm that connects all the components effectively to offer certain location-based services and applications [10]. This section will describe the fundamental of a LBS architecture including the basic components of the LBS and the request-response communication paradigm. The limitations of the current LBS framework will be identified, providing the rationale for a new LBS framework

2.1. LBS components

A Location Based Service (LBS) system generally consists of several components. The work [2] describes a LBS with four key components that are common to all applications: mobile device, content provider, communication networks and positioning systems. A LBS model of six components, mobile devices, content provider, application provider, communication network, positioning technology and user, were outlined in [3]. The 'mobile devices' are referred to electronic devices, such as smartphones and laptops that are used to communicate with the LBS service, which is capable of connecting to a mobile cellular network and transferring voice and/or data. It is essential that these mobile devices include positioning capability. It is desirable that the device can host an operating system, such as iOS or Android for various mobile apps. The 'content provider' creates host media (contents) in terms of geographic location and related data from different sources that are provided to mobile devices, either reactively or pro-actively. The 'application and service providers' provide various services based on the user requests. The 'communication network' in [3] refers to the communication channels that users (mobile devices) can send requests and receives the information, which includes voice and data to/from LBS services (service providers) across mobile networks. Positioning technology is a hybrid positioning technology embedded in each mobile device that is used to determine and/or track client mobile devices. The dominant positioning technology used for LBS are GNSS, Cell-ID and Cell tower triangulation. The user refers to a client or a person who is using mobile devices to send the requests to LBS services (service providers) and receive the responses from the service providers through communication networks. While the user and mobile device are in the mobile device end, the 'Service and Application Provider' and the 'Data and Content Provider' might be the same actor in the LBS architecture. The LBS provides a versatile structure that enables various applications to be implemented by changing the contents and apps of mobile devices. Figure 1 shows the six major components of the LBS required to deliver fully functional services.

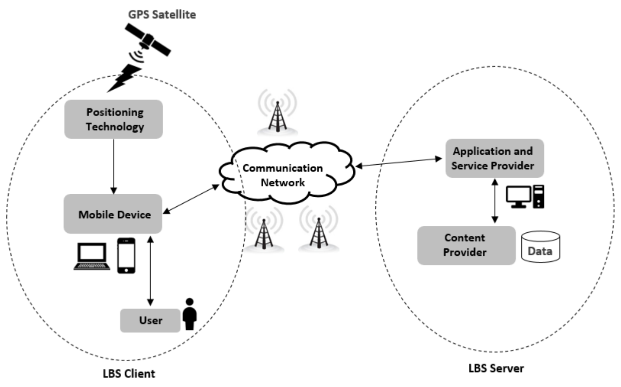

Figure 1. Client-server LBS architecture and six components

2.2. The LBS Request-Response communication paradigm

The LBS architecture fundamentally consists of three-layers: Client, Server and Wireless Communication networks[11]. Each layer contains a member of sub-functions as follows:

- Client: request, pull, display, and interface.
- Server: response, push, collecting, computing, management, contents, and services.
- Wireless Communication: transmission, connection, services, real-time, protocols, and interaction.

Referring to Figure 1, the user initially sends a service request with a current location acquired from the positioning component to service providers through the mobile application across a communication network. When the service provider receives a user request, it will be sent to a content provider to provide the requested information to a user. The above LBS client-server connection utilizes the Request-Response model. The Request-Response paradigm enables bidirectional communication between the server and a client. As shown in Figure 2 for bidirectional communication between endpoints, the initiator of the communication endpoint A sends a request message, and the target endpoint B receives the request and processes it and then sends a response message to the original initiator A. Many LBS deployments follow the same client-server architecture, or implement an interactive communication, that is, both endpoints have information to send to the other side, the receipt of information are fully acknowledged. However, not all LBS deployments need to have the same characteristics. For instance in some scenarios, one-way communication from a mobile device to a server or a server to a device is needed.

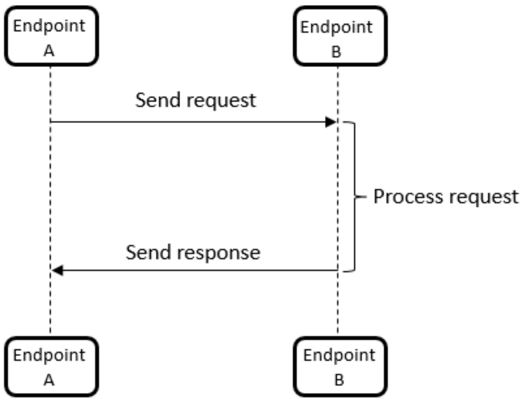

Figure 2. Request-Response paradigm

2.3. Limitations of client-server architecture and current LBS services

The client-server architecture is a centralized model, in which clients requests are sent to the server to receive the information. Such client-server architecture performs well when data is stored in a central server, such as a file and a database server, or contents are accessible from third parties through the server. With the client-server model, a single server can accommodate various services to many client requests and the client can send requests to multiple servers through a network. The client-server architecture consists of three major components including hardware, software, and communication middle-ware, which can communicate with each other. Each component of the client-server comprises: (1) hardware is related to the client and the server that client usually requests the services to the server, which is the computer that provides the services and the responses to every client requests; (2) software is used to response the requirement of users (clients); (3) communication middle-ware is used to transmit information among the client and the server across a network [12].

However, the client-server architecture has some disadvantages. The server side will take more overheads to serve thousands of simultaneous client requests, due to the Request-Response communication. Figure 3 shows the centralised LBS structure. Every HTTP request from the client will have to create a new TCP connection, and the client has to wait until the responses to the previous HTTP requests have been completed [13]. These overheads lead to the reduced performance of the system, such as high latency. In addition, the client-server architecture has a scalability problem in a large scale network. Normally, more servers are introduced to meet the demand of a large number of clients. Because of these problems, the client-server architecture is not considered suitable for supporting Device-to-Device (D2D) LBS applications in terms of scalability and timeliness.

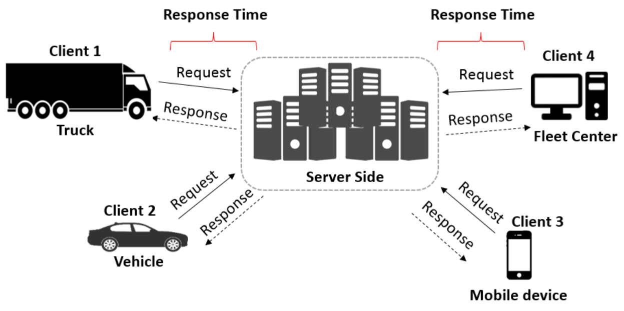

Figure 3. Centralised client-server Location-based services

3. DEVICE-TO-DEVICE LBS ARCHITECTURE BASED ON THE PUBLISH-SUBSCRIBE PARADIGM

To overcome the limitations of the client-server LBS architecture and difficulties in the current LBS deployments, we propose the Device-to-Device (D2D) LBS concept based on internet connections. Specifically, this is to introduce the publish-subscribe communication paradigm in the application layer to support the proposed D2D LBSs with the mobile broadband services (MBB) under the current 4G-LTE networks and the future 5G extreme Mobile Broadband (eMBB) services. In the following, the new D2D LBS architecture is outlined, followed by several use cases and requirements for communications.

3.1. Device-to-Device LBS use cases

The term "Device-to-Device (D2D)" refers to direct communication between two mobile devices. D2D was initially proposed as a new paradigm in cellular networks. LTE-Direct is an innovative D2D technology that enables mobile devices and applications to passively discover and interact with the world around them in a privacy sensitive and power efficient manner [14]. LTE-Direct creates new proximity service opportunities for the entire mobile industry in social networking, venue services, loyalty services, local advertising, and much more. Along the same line, LTE-Vehicle enables Vehicle-to-Everything (V2X) communication and is considered to be one of the optimal choices for effective connected vehicles. On the other hand, the 5G uMTC services provide ultra-reliable and low latency communication for high demanding applications. V2X is one of most important applications of uMTC services in the future. LTE-Direct, LTE-Vehicle and 5G uMTC services will certainly enhance the LBS applications and empower many new LBS use cases. However, in the D2D LBS concept, devices are IP-based, generally including smartphones, computers, mobile devices on-board vehicles, roadside devices, network servers and Internet of Things (IoT) end-devices. For LBS applications, all devices have location sensors. For the connected vehicle applications, D2D connections can mean vehicle-to-vehicle, vehicle-to-infrastructure , vehicle-to-network and vehicle-to-people. More specifically, the D2D LBS can support, but not to limit to the following use cases:

- V2V and V2I safety message and location data exchanges for cooperative awareness. The data are required to be transmitted at the rate between 1 and 10 Hz. The payload of each message ranges from 60 to 1500 Bytes [15], affected by the road and traffic conditions. The message types may include GNSS measurements in the standards of Radio Technical Commission for Maritime Services (RTCM) and the SAE J2735 Message Set Dictionary, and the emerging SAE J2945.1 Communication Minimum Performance Requirements standard [16]. While the Basic Safety Message (BSM) (Part-1) in the SAE J2735 is less than 60 Bytes, the optional BSM including RTCM corrections can range from 300 to 1200 Bytes for 1 to 4 GNSS constellations.
- V2N and V2I traffic data exchanges for traffic efficiency. This transmission does not have strict delay latency and reliability requirements, as prompt action at the vehicle side is not neccessarily required. Each vehicle updates the Traffic Management server (uplink) every few seconds with location, status and road information for the more efficient route selection. The payload of this

type of message is up to 1500 Bytes [15]. The response from the Traffic Management servers (downlink) includes digital map title updates, which may be in the size of a few MBytes, but the transmission is event-driven and is not time critical.

- V2P and P2V data exchanges for vulnerable road users (VRUs). This use case is similar to the cooperative awareness category in terms of latency and reliability requirements. The difference is in that the destination device is a user equipment (e.g., smartphone), where the needed information is less, e.g., payload sizes from 60 to 120 Bytes [15].
- Location-based vehicle and personal tracking. This offers a tracking platform within a group of people and vehicles in the same company without a centralised LBS server. In other words, a mobile app can be designed to track family members. The data exchange rate of 1 Hz is sufficiently frequent and the message size should be about 60 to 120 Bytes.
- Device to Server GNSS raw data collections and Server to Device distribution in RTCM formats. This is similar to the V2V RTCM data exchanges, but the data are collected or transmitted at the frequency of 1 Hz or lower. If the payload of each message includes all raw measurements from multiple constellations, the message size can be 1200 to 1500 Bytes.

Overall, in the above D2D LBS use cases, the messages of 60 to 1500 Bytes are required to be transmitted at the frequency of up to 10 Hz. While the LTE-Direct and 5G and uMTC technologies may support all use cases of D2D LBSs in years of the future, many D2D LBS use cases may also be deployed right now through the application protocols based on the publish-subscribe communication paradigm. Therefore, in the following subsections, we introduce the publish-subscribe communication paradigm, and propose the D2D LBS framework, which can be implemented with the current Internet connections.

3.2. Publish-Subscribe paradigm and D2D connections

The publish-subscribe paradigm enables unidirectional communication from a publisher to one or more subscribers. In software architecture, publish-subscribe is a messaging pattern where senders of messages, are called publishers, and receivers are called subscribers. Publishers do not program the messages to be sent directly to specific subscribers, but categorize published messages into classes, in which subscribers express interest in one or more classes and receive messages that are of their interest. Publishers and subscribers communicate for the interested messages without knowledge of subscribers or publishers. The Publish-subscribe model enables event-driven architectures and asynchronous event notifications, while improving performance, reliability and scalability. Figure 4 illustrates the publish-subscribe model with three publishers and three subscribers for the commonly interested messages of mobile phone, vehicles and road information. Vehicle users declare their interests in both roadside data and vehicles data to the publisher-subscriber server. When the server has new data available in that topic, the server platform pushes the data or messages to interested vehicle subscribers. This process will repeat for all publishers and subscribers.

The publish-subscribe model is well suited for many D2D LBS deployments as a device can be both a publisher and subscriber, and can communicate with each other through the publisher-subscriber server. D2D LBS can benefit from the loose coupling between communication endpoints, low latency and better scalability. The parallelism and multicast capabilities of the underlying transport network can be improved by a publish-subscribe model. This model can also support point-to-multipoint and multipoint-to-multipoint. In other words, one device can choose to receive data from multiple devices or data sources. Although a server is needed for exchange of data, it can be set close to data sources and many can be used in the distributed manner.

In the publish-subscribe model, subscribers typically receive only a subset of the total messages published. The process of selecting messages for reception and processing is called *filtering*. Three publish-subscribe schemes, including topic-based, content-based and type-based, have been researched to identify the events of interest [17]. How these schemes may be implemented for various location-based V2X applications is an important issue, deserving dedicated research attentions. However, research on geographic routing (GR) or position-based routing protocols for vehicular ad-hoc networks (VANET) [18] can serve as a good reference for the design of the above filter alongside the application layer protocols.

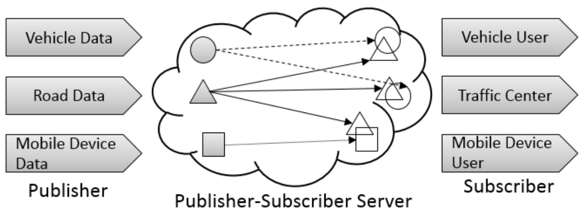

Figure 4. Illustration of the publish-subscribe model with three publishers and three subscribers.

With various publish-subscribe schemes, flexibility of contents and dynamic contents are offered in order to exchange data between large numbers of entities without establishing any contracts or knowing each other [19]. However how to implement a filtering scheme to effectively support a particular D2D LBS use case is a new research problem that requires specific attention in due course.

3.3. Device-to-Device LBS architecture with the publish-subscribe communication paradigm

Based on the publish-subscribe paradigm, the LBS reference architecture must be updated. Figure 5 is a diagram of a D2D LBS architecture that allows clients to exchange data each other, depending on the interests to particular classes of data. Each client is both a publisher and subscriber. As a result, for instance, Clients 1, 4 and 5 are connected through the server and can exchange data. Clients 2 and 3 can exchange data. It must be noted that a client may still be connected to a centralised LBS server in the client-server model. The D2D LBS architecture adds supplementary use cases rather than replaces the existing deployments. With the new LBS architecture, the scope of LBS is extended to support D2D services as well as the communication connections or simultaneous mobile users in the services can be extended.

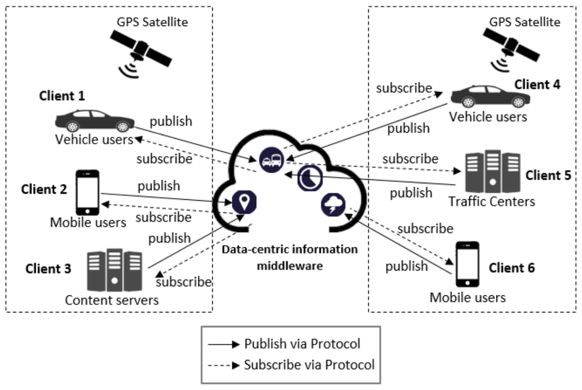

Figure 5. Diagram of Device-to-Device LBS architecture with publish-subscribe communication paradigm.

4. APPLICATION PROTOCOLS SUPPORTING PUBLISH-SUBSCRIBE PARADIGM

In the application layer of the TCP/IP layered protocol model, Application protocols are responsible for handling the communications between application entities, such as devices, gateways, and Applications [20]. They typically support the flow of data (eg., readings and measurements) from devices to a application platform and the flow of command or control information in the reverse direction. They support different communication patterns and enable varying paradigms of interaction between applications and devices. There have been many application protocols developed over years. In this section, we introduce two IP-based protocols, which could be potentially very suitable for D2D LBS applications.

4.1. Data Distribution Service for Real-Time Systems

The Data Distribution Service for real-time systems (DDS) is an Object Management Group (OMG) Machine-to-Machine (sometimes called middle-ware) standard that aims to enable scalable, real-time, dependable, high-performance and interoperable data exchanges using a publish-subscribe pattern [21]. DDS can integrate massive information networks while maintaining performance of the real-time services, including low latency and high throughput as well as to minimize network resources consumption in a dynamic and independent environment, including wireless and satellite connections [22]. DDS enables reliable communication with more than 20 Quality of Services (QoS) policies, such as reliability, durability and priority, which regards to various objectives of communication. The DDS architecture has divided into two layers, which include the Data-Centric Publish-Subscribe (DCPS) and the Data Local Reconstruction Layer (DLRL). The DCPS enables the delivery efficiency of data from the publishers to subscribers [23]. The DLRL is an optional platform for providing an object-oriented view of DCPS. DDS is based on the idea of a Global Data Space (GDS) where the publishers and subscribers can exchange information. DDS is completely distributed communication. The GDS is used to identify circulating information in the system [21]. As shown in Figure 6, the POI, road and traffic information providers and vehicle and mobile users can interact by identifying themselves on a network. Then, the data from publishers will be delivered to the subscribers, who are subscribing the same topic via the DDS Global Data Space (data bus). With a decentralized architecture of DDS technology, publishers and subscribers can communicate directly without broker (broker-less) through the Real-Time Publish-Subscribe (RTPS) protocol, which is the standard network protocol defined by the OMG, for supporting multi-cast communication and connectionless best-effort transport layer, such as UDP/IP [24]. DDS is mostly used for the high-performance systems (low latency, real-time, and well scalability), mission-critical systems via distributed data and event-driven processing. Various DDS middleware have been developed by vendors such as RTI DDS and open-source software development such as OpenSplice DDS and OpenDDS. In our evaluation for D2D LBS applications in the next section, we choose the widely-used open source OpenDDS.

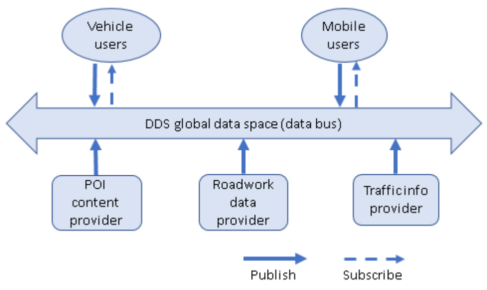

Figure 6. The publish-subscribe model of DDS.

4.2. Message Queue Telemetry Transport (MQTT) protocol

The Message Queue Telemetry Transport (MQTT) protocol has been developed by IBM and standardized in 2013 by OASIS [25]. MQTT is a message-centric wire protocol designed for Machine-to-Machine (M2M) communications that provides lightweight, simple implementation, open standard, reliability, and efficiency with regards to processor, memory, and network resources. The publish-subscribe communication model is used to transfer the telemetry-data in the messages format from publisher (device), along restricted environments and unreliable networks, to brokers across TCP/IP. Recently, the two main versions of MQTT are mentioned as follows: (i) MQTT version 3.1 has been developed to transmit data over Transmission Control Protocol/Internet Protocol (TCP/IP) and it is defined as the basic transport and network service and (ii) MQTT-SN which has been developed for transmitting data through User Datagram Protocol (UDP) over low-bandwidth wireless communication networks [26], [27]. MQTT has defined three QoS levels for data delivery. In the

QoS level 0, a message is sent only once, which means that the message can be lost, and there is no guarantee that the message will arrive to the destination. In the QoS level 1, a message is sent at least once, and it will be retransmitted until a broker receives acknowledgement from subscriber by using PUBACK. As a result, a subscriber could receive message multiple times due to the message retransmission. In the QoS level 2, a message is sent exactly once by using a four-way handshake which guarantees that the message will not be lost. However, the end-to-end delays will be increased as more packets are exchanged. For D2D LBS use cases, QoS 1 is the default mode of data transfer. The several MQTT broker products have been used in a worldwide not only industries but also academic research. The benchmark evaluation of MQTT scalability has been revealed that the performance in term of latency, CPU and message rate of JoramMQ and Mosquitto is desirable and satisfiable [28]. In the next section, the D2D LBS applications will be evaluated using an open source Mosquitto MQTT broker [29]. Figure 7 illustrates the basic model of MQTT. Traffic, road and POI data publishers publish

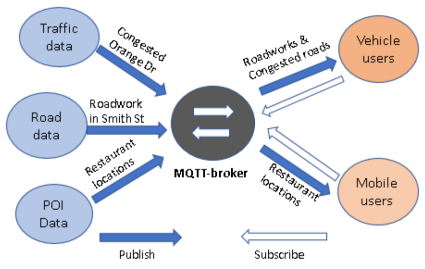

Figure 7. The publish-subscribe model of MQTT.

messages with specifying the topic names to a central broker (server). When the broker receives the published messages from the publishers, it then transmits (publishes) these messages to the mobile and vehicle subscribers based on their subscription topics. MQTT has been widely adopted in various IoT applications, such as home automation, medical applications and remote monitoring as it can offer the implementation on limited memory or power devices and in constrained networks, such as low bandwidth, high latency, and fragile connection.

4.3. Comparison between the application protocols

MQTT is a lightweight messaging protocol that has been designed for highly resource-constrained devices. It is a message-centric approach that focuses on data transmission without regard to the contents. DDS, on the other hand, is a data-centric approach that focuses on data structure which has to be understood by the middleware before exchanging information. It requires more computing power for running the DDS middleware and is suitable for complex applications by providing a rich set of QoS policies and hard real-time. Both DDS and MQTT deliver real-time communication as well as maintaining QoS. For D2D LBS applications, adoption of DDS or MQTT protocols depend on the performance requirements, network and computing resources available for the use cases. In the following section, we examine the latency performance of the OpenDDS and the feasibility of MQTT based on Mosquitto protocols for the V2V use cases, which requires light-weight data exchanges.

5. D2D DATA EXCHANGE PERFORMANCE WITH MQTT and DDS PROTOCOLS

To verify the concept of the D2D LBS based the publish-subscribe model, we perform two experiments with the MQTT and DDS protocols in device-to-server and device-to-device data exchange. The objectives of both experiments are to find the data exchange Round-trip time (RTT) and the packet-loss rate (PLR) between multiple publishers-subscribers at different message sizes. The experiments and results are presented in the following two subsections.

5.1. Experiment 1: Static tests with OpenDDS protocols and RTT and PLR results

The experiment used two 4G mobile phones for Internet connections and two PCs to run OpenDDS protocols, in order to simulate vehicle communications. One PC was running a 4 core AMD processor at 4 GHz with 16 GB of RAM while the other PC was equipped with an 8th generation Intel CPU with 6 cores running at 4.8 GHz with 16 GB of RAM. There is no broker between two PCs. OpenDDS operates in the RELIABLE QoS mode. Each PC sets 4pubs to 4subs to simulate the connections of the host vehicle and 4 target vehicles. The message size settings include 50, 100, 300, 500, 1000, 2000 Bytes, to reflect different V2X message size requirements as discussed in Section 3.1. Each test sets the transmission frequency of 10 Hz and runs for 120 seconds. For the 4pubs to 4subs, there are a total of 4800 messages to transmit and receive. Table 1 lists the performance parameters such as the average and maximum RTT, the delayed message rates beyond the bounds of 100 ms, packet-loss rate for the lost messages and the total packet-loss rates beyond the 100 ms bound. Fig 8 plots the RTT records for transmission at the frequency of 10 Hz and the message size of 1000 Bytes over all the messages received. It is observed that when the message size increases from 50 to 2000 Bytes, the percentages of the delayed messages over 100 ms increase from 0.85% to 5.29%. The total PLRs over the limit of 100 ms remain in the range of 5.25% to 8.76%. This result shows the good potential for using OpenDDS to support moderate demanding V2X communications at the frequency of 10 Hz and up to 2000 Bytes per message.

Table 1. RTT and PLR results from static OpenDDS tests

Message Size (Bytes)	Average RTT (ms)	Maximum RTT (ms)	Messages transmitted	Delayed message beyond 100 ms	PLR	Total PLR beyond 100 ms
50	74.11	229.52	4493.00	0.85%	6.40%	7.25%
100	72.66	234.18	4544.00	1.36%	5.40%	6.76%
300	75.80	218.15	4632.00	2.76%	3.50%	5.26%
500	77.12	531.68	4608.00	3.91%	4.00%	7.91%
1000	78.59	288.52	4669.00	5.29%	2.70%	7.99%
2000	82.97	338.57	4590.00	4.36%	4.40%	8.76%

Table 2. RTT and PLR results from static MQTT tests with 4G-LTE connections

Message size (Bytes)	frequency (Hz)	Pub/Sub	Average RTT (ms)	Maximum RTT (ms)	Delayed message (100 ms)	PLR
100	1	1	57.99	203.00	0	0.00%
100	1	4	60.15	1297.00	20	0.21%
100	1	10	77.73	1467.00	90	0.15%
100	10	1	45.46	1034.00	62	1.03%
100	10	4	46.73	1125.00	918	0.96%
100	10	10	47.73	2262.00	3217	0.54%
600	1	1	65.98	390.00	0	0.00%
600	1	4	58.37	1171.00	16	0.17%
600	1	10	77.57	1660.00	108	0.18%
600	10	1	46.93	260.00	21	0.35%
600	10	4	53.18	1210.00	375	0.39%
600	10	10	46.58	1591.00	2574	0.43%
1000	1	1	55.93	110.00	0	0.00%
1000	1	4	59.39	1160.00	20	0.21%
1000	1	10	93.24	1376.00	100	0.17%
1000	10	1	49.97	280.00	19	0.32%
1000	10	4	56.86	1230.00	652	0.68%
1000	10	10	58.55	1881.00	28477	4.75%
2000	1	1	57.34	137.00	0	0.00%
2000	1	4	61.47	1271.00	16	0.17%
2000	1	10	71.78	1232.00	110	0.18%
2000	10	1	43.85	250.00	5	0.08%
2000	10	4	49.96	1700.00	1887	1.97%
2000	10	10	60.53	5569.00	36796	6.13%

5.2. Experiment 2: Static and dynamic tests with Mosquitto and RTT and PLR results

The static experiments used the same two laptops. The first one was placed in an office room and connected to the Ethernet. The second one was connected to the Internet with a 4GX Wi-Fi modem. Both laptops ran the middleware, Mosquitto in the default QoS 1 mode. The first laptop operated

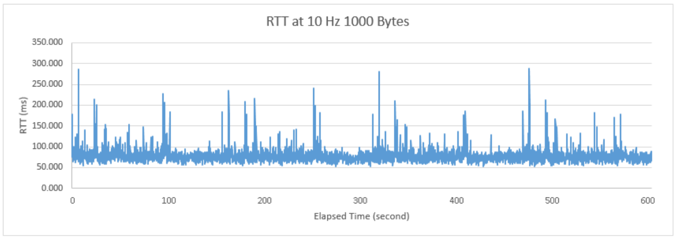

Figure 8. RTT measurements for 4pubs-4subs connections from static OpenDDS testing

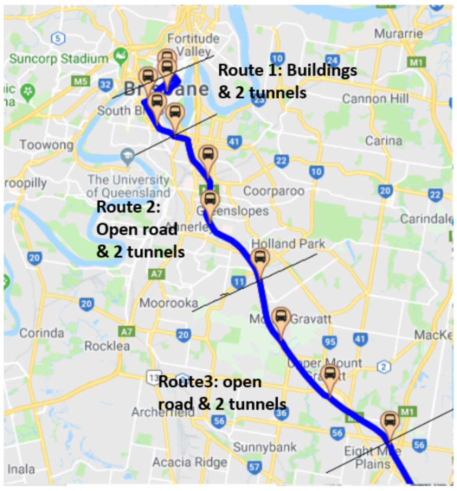
Route 1: Buildings & 2 tunnels
Route 2: Open road & 2 tunnels
Route3: open road & 2 tunnels

Figure 9. Illustration of the bus routes in dynamic MQTT testing

as a MQTT broker. The second laptop was used as a mobile device. The settings for publishers and subscribers included 1pub/1sub, 4pubs/4subs and 10pubs/10subs to simulate the connections between 1, 4 and 10 target vehicles. The message size settings included 100, 600, 1000, 2000 Bytes, to reflect different V2X requirements. The transmission frequency was set 1 and 10 Hz and each test ran for 10 minutes. RTT is the time delay measured between the time of transmission messages from a publisher and the time of receiving the same message at a subscriber. A timestamp was added to each message published and received from the mobile device so that the RTT data can be measured.

Table 2 summarizes the average and maximum RTTs in milliseconds, the number of the delayed messages beyond 100 ms, and total PLR beyond 100 ms. The results show that the average RTT of lower than 100 milliseconds is achieved by the default mode QoS 1 at the frequency of 10 Hz for the message size of 100 to 2000 Bytes and in three publisher/subscriber settings. This finding is similar to the OpenDDS testing results. For the setting of 4pubs/4subs, the results show that at the data transfer frequency of 10 Hz, the PLR is 4.75%. It is clearly observed that overall the MQTT offers slightly lower RTT and PLRs for the same pub/sub setting as for OpenDDS.

In the dynamic testing scenario, the broker is set with a cloud virtual machine. Two sets of laptops and 4G modems for mobile broadband (mBB) are carried on a bus that travels at the speeds of 60 Km/h to 80 Km/h on a Busway for about 35 minutes. The busway includes several poor 4G reception areas, mainly due to high building structures and tunnels. Figure 9 illustrates three parts of the bus route, and the travel/stop time is about 11-12 minutes each. Two laptops/4G modems operate simultaneously and send and receive the messages of 100 and 1000 Bytes, respectively, but all at the frequency of 10 Hz. Figures 10 to 12 show the RTT in millisecond for the 1pub/1sub, 1pub/4subs, 1pub/10subs settings. For each setting, Table 3 outlines the average RTT, the delayed messages and PLR results between 100 ms and 1000 ms and between 200 ms and 1000 ms. Any RTT delays over the threshold of 1000 ms are grouped into RTT outages and are removed from the RTT averaging. The results show that average RTTs are lower than 100 ms with the message sizes of 100 and 1000 Bytes after removing the RTT outages. The outages are considered being caused by 4G signal reception coverage or degradation problems due to cell handover and physical structures such as tunnels etc. Generally, RTT and PLR performance depend on user mobility, message size and number of connected vehicles under normal 4G signal reception conditions. Specifically for the message size of 100 Bytes, the PLRs over delays of 200 to 1000 ms are 0.38% and 3.58%, for the settings of 1pub/1sub and 1pub/10subs, respectively. On the other hand, for the message size of 1000 Bytes, the PLRs over delays of 200 to 1000 ms are 0.63% and 5.77% for the settings of 1pub/1sub and 1pubs/10subs, respectively. Bus dynamic tests show the realistic performance of MQTT for V2X data exchanges at the latency of 100 ms and 200 ms.

The average RTT delays in the above testing are less than 100 ms, which is much lower than the average RTT delays of 300-400 ms obtained by [30] in their running vehicle 4G-MBB tests with respect to the vehicle running velocity of 60 to 120 Km/h. Similarly, we do not conclude that 4G-MBB can meet the lowest RTT requirement of 100 ms for safety applications. However, our MQTT test results do show that 4G-MBB meets the lower RTT requirements of 200 ms at the PLRs of less than 6%, when a host vehicle connects to 10 vehicles. Comparing to DSRC for V2X communications, LTE based MQTT is less-favourable in terms of the latency performance, but its PLR performance does not depend on the distance between vehicles on roads. This performance allows a considerable number of medium-demanding V2X safety applications, especially V2I and V2P use cases to be supported by 4G-LTE by adopting the MQTT protocols in the application layer.

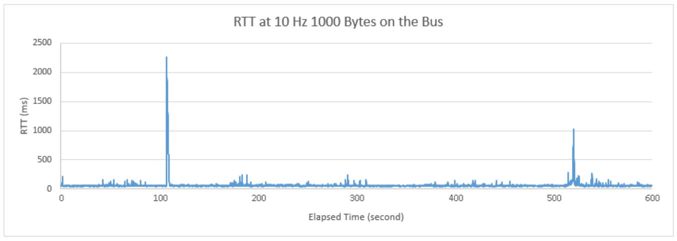

Figure 10. RTT measurements for 1pub/1sub connections from dynamic MQTT testing

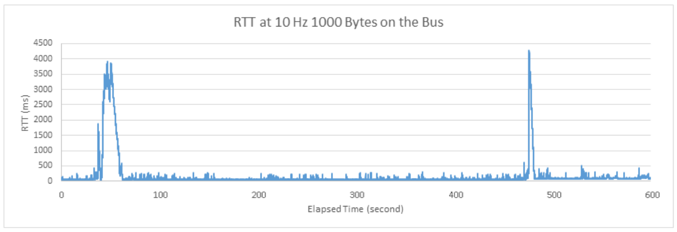

Figure 11. RTT measurements for 1pub/4subs connections from dynamic MQTT testing

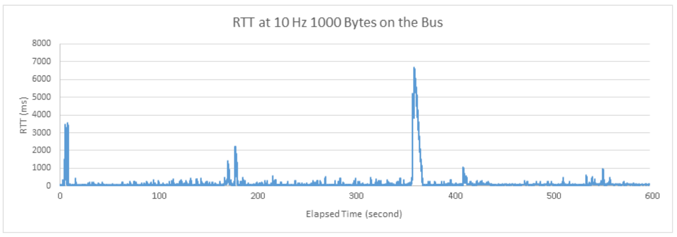

Figure 12. RTT measurements for 1pub/10subs connections from dynamic MQTT testing

6. Conclusion

Location-Based Services (LBS) deployments adopt the client-server architecture in the connected vehicle applications such as road personal navigation, tracking or fleet management and augmented reality. But the client-server architecture is a centralised structure, and it is tightly coupling with space,

Table 3. RTT, delayed messages and PLR obtained with a testing vehicle running at 60-80 kph

Message size (Bytes)	Pub/Sub	Total record	Average RTT (ms)	PLR (100 - 1000 ms)	PLR (200 - 1000 ms)	Outages (>1000 ms)	PLR (>1000 ms)	4G signal reception
100	1P1S	6000	67.88	1.07%	0.38%	44	0.73%	Route 1
100	1P4S	24000	50.58	2.03%	0.64%	0	0.00%	Route 2
100	1P10S	60000	63.43	6.60%	3.57%	0	0.00%	Route 3
1000	1P1S	6000	59.79	2.40%	0.63%	19	0.32%	Route 1
1000	1P4S	24000	79.84	13.40%	3.58%	908	3.78%	Route 2
1000	1P10S	60000	90.85	14.97%	5.77%	1417	2.36%	Route 3

* PLR (100 - 1000 ms) = 100 ms <RTT<= 1000 ms, PLR (>1000 ms) = RTT >1000 ms,
PLR (200 - 1000 ms) = 200 ms <RTT<= 1000 ms

structure and time constraints and has limitations in support of Device to Device (D2D) applications in terms of low latency, high reliability and large scalability performance. These are important requirements for high-demanding connected vehicle applications, such as Vehicle-to-Everything (V2X) applications. Currently, dedicated short-range communication (DSRC) and 4G-LTE are two widely used candidate schemes for connected vehicle applications. However, the recent 4G-LTE experimental results have shown that the average RTTs are 300-400 ms for the vehicle speeds from 60 Km/h to 120 Km/h. Thus, 4G-LTE cannot meet the lowest RTT requirement of 100 ms for safety applications. DSRC latency can be well within the lowest requirement of 100 ms, but their PLRs are significantly degraded when the inter-vehicle distances are over 200 m. In this contribution, a D2D LBS framework based on the publish-subscribe architecture has been proposed. Publishers and subscribers communicate data of interest or messages without knowledge of subscribers or publishers. The publish-subscribe model enables event-driven architectures and asynchronous event notifications, while improving performance, reliability and scalability. The publish-subscribe model can support many D2D LBS deployments as a device can be both a publisher and subscriber, and can communicate with each other through a publisher-subscriber server. D2D LBS can benefit from the loose coupling between communication endpoints, lower latency and higher scalability. The paper also introduced two well-established publish-subscribe application protocols: Distributed Data services (DDS) and Message Queue Telemetry Transport (MQTT) for implementation of the proposed D2D LBSs.

To demonstrate the concept and benefits of the D2D LBS based on the publish-subscribe model for connected vehicle applications, DDS and MQTT tests were designed to assess the round-trip time (RTT) and packet-loss rates (PLR) in the sense of 100 ms and 200 ms latency under a single 4G connection. Static and dynamic experimental results generally show that the average RTT of less than 100 ms can be achieved with tested DDS and MQTT protocols for D2D data exchanges at the frequency of 10 Hz and messages sizes of 100 to 2000 Bytes. The packet-loss rates however depend on user mobility, the message sizes and number of connected vehicles, but typically for the message sizes of 100 and 1000 Bytes and connecting 10 vehicles, the PLRs within 200 ms and 1000 ms are 1% and 6%, respectively. PLRs could be partially caused by RTT outages over the 4G poor receptions areas, but not due to use of the application protocols. Generally, it is feasible to use DDS and MQTT protocols to implement high density, medium-demanding D2D LBS use cases, such as Vehicle to People (V2P) and Vehicle-to-Infrastructure (V2I) safety applications. As the DDS and MQTT middle-ware codes can be implemented in the application layer, the finding is promising: many V2X applications could be implemented as a mobile apps with the current 4G mobile broadband (MBB) services and future 5G eMBB services.

Acknowledgments: This work was partially supported by the Australian Research Council under the linkage grant LP150101172. The first author also acknowledges the scholarship support of Thailand Government and Queensland University of Technology.

References

1. Steiniger, S.; Neun, M.; Edwardes, A.; Lenz, B. Lecture notes: Foundations of Location Based Services, 2006. Department of Geography, University of Zurich.

2. Ferraro, R.; Aktihanoglu, M. *Location-Aware Applications*; Manning Publications Co.: Greenwich, CT, USA, 2011.
3. Buczkowski, A. Location-based Marketing: the academic framework, Feb. 2012. Masters Program in Geospatial Technologies, pp. 1-77.
4. Michael, G.E.K.; Michael, M. The social and behavioural implications of location-based services. *Journal of Location Based Services* **2011**, *5*, 121–137.
5. Huang, B.; Gao, Y. Integrated Indoor Positioning with Mobile Devices for Location-Based Service Applications. Database Systems for Advanced Applications; Han, W.S.; Lee, M.L.; Muliantara, A.; Sanjaya, N.A.; Thalheim, B.; Zhou, S., Eds.; Springer Berlin Heidelberg: Berlin, Heidelberg, 2014; pp. 329–341.
6. Bergenhem, C.; Hedin, E.; Skarin, D.; others. Vehicle-to-vehicle communication for a platooning system. *Procedia-Social and Behavioral Sciences* **2012**, *48*, 1222–1233.
7. Wu, W.; Yang, Z.; Li, K. Internet of Vehicles and applications. In *Internet of Things*; Buyya, R.; Dastjerdi, A.V., Eds.; Elsevier, 2016; pp. 299 – 317.
8. Huang, J.M. Research on internet of vehicles and its application in intelligent transportation. Applied Mechanics and Materials. Trans Tech Publ, 2013, Vol. 321, pp. 2818–2821.
9. Contreras, J.; Zeadally, S.; Guerrero-Ibanez, J.A. Internet of Vehicles: Architecture, Protocols, and Security. *IEEE Internet of Things Journal* **2017**, pp. 1–9.
10. Feng, Y.; Wang, C. A Unidirectional Communication Architecture for Extended Location-Based Services. China Satellite Navigation Conference (CSNC) 2018 Proceedings; Sun, J.; Yang, C.; Guo, S., Eds.; Springer Singapore: Singapore, 2018; pp. 267–283.
11. Pontikakos, C.; Sambrakos, M.; Glezakos, T.; Tsiligiridis, T. Location-based services: A framework for an architecture design **2006**. *14*, 273.
12. David Barkai. An introduction to peer-to-peer computing. *Intel Developer update magazine* **2000**, pp. 1–7.
13. de Saxcé, H.; Oprescu, I.; Chen, Y. Is HTTP/2 really faster than HTTP/1.1? 2015 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2015, pp. 293–299.
14. Wang, Y.; Wei, L.; Vasilakos, A.V.; Jin, Q. Device-to-Device based mobile social networking in proximity (MSNP) on smartphones: Framework, challenges and prototype. *Future Generation Computer Systems* **2017**, *74*, 241 – 253. doi:https://doi.org/10.1016/j.future.2015.10.020.
15. Boban, M.; Kousaridas, A.; Manolakis, K.; Eichinger, J.; Xu, W. Use Cases, Requirements, and Design Considerations for 5G V2X. *preprint arXiv:1712.01754* **2017**.
16. Kenney, J.B. Dedicated Short-Range Communications (DSRC) Standards in the United States. *Proceedings of the IEEE* **2011**, *99*, 1162–1182.
17. Eugster, P.T.; Felber, P.A.; Guerraoui, R.; Kermarrec, A.M. The Many Faces of Publish/Subscribe. *ACM Comput. Surv.* **2003**, *35*, 114–131.
18. Boussoufa-Lahlah, S.; Semchedine, F.; Bouallouche-Medjkoune, L. Geographic routing protocols for Vehicular Ad hoc NETworks (VANETs): A survey. *Vehicular Communications* **2018**, *11*, 20 – 31.
19. Ion, M.; Russello, G.; Crispo, B. Supporting Publication and Subscription Confidentiality in Pub/Sub Networks. Security and Privacy in Communication Networks; Jajodia, S.; Zhou, J., Eds.; Springer Berlin Heidelberg: Berlin, Heidelberg, 2010; pp. 272–289.
20. Rayes, A.; Samer, S. *Internet of Things From Hype to Reality: The Road to Digitization*; Springer, 2016.
21. Bellavista, P.; Corradi, A.; Foschini, L.; Pernafini, A. Data Distribution Service (DDS): A performance comparison of OpenSplice and RTI implementations. IEEE Symposium on Computers and Communications (ISCC), 2013, pp. 000377–000383.
22. Franklin, S.J.; Do, Q.V. An investigation of data distribution services (DDS) performance and specification using a SysML profile. Systems Engineering Test and Evaluation (SETE"11), 2011, pp. 1–8. Adelaide, Australia.
23. Esposito, C.; Russo, S.; Crescenzo, D.D. Performance assessment of OMG compliant data distribution middleware. 2008 IEEE International Symposium on Parallel and Distributed Processing, 2008, pp. 1–8.
24. OMG (Object Management Group). The Real-Time Publish-Subscribe Wire Protocol DDS Interoperability Wire Protocol Specification, 2014. OMG Document, V2.2.

25. Masek, P.; Hosek, J.; Zeman, K.; Stusek, M.; Kovac, D.; Cika, P.; Masek, J.; Andreev, S.; Kröpfl, F. Implementation of True IoT Vision: Survey on Enabling Protocols and Hands-On Experience. *International Journal of Distributed Sensor Networks* **2016**, *12*, 8160282.
26. Stanford-Clark, A.J.; Wightwick, G.R. The Application of Publish/Subscribe Messaging to Environmental, Monitoring, and Control Systems. *IBM J. Res. Dev.* **2010**, *54*, 396–402.
27. Pereira, C.; Aguiar, A. Towards Efficient Mobile M2M Communications: Survey and Open Challenges. *Sensors* **2014**, *14*, 19582–19608.
28. Scalagent. Benchmark of MQTT servers. Available online: http://www.scalagent.com/IMG/pdf/Benchmark_MQTT_servers-v1-1.pdf, (accessed on 17 November 2018).
29. Light, R.A. Mosquitto: server and client implementation of the MQTT protocol. *The Journal of Open Source Software* **2017**, *2*. 265, doi:10.21105/joss.00265.
30. Xu, Z.; Li, X.; Zhao, X.; Zhang, M.H.; Wang, Z. DSRC versus 4G-LTE for connected vehicle applications: A study on field experiments of vehicular communication performance. *Journal of Advanced Transportation*, **2017**, *2017*. 10 pages, doi:org/10.1155/2017/2750452.