Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 56 (2015) 349 – 356

Procedia Computer Science

The 10th International Conference on Future Networks and Communications (FNC-2015)

Real-Time QoS-Aware Vehicle Tracking: An Experimental and Comparative Study

Basem Almadani*, Abdullah Al Mamun**, Ahmad Khayyat

Department of Computer Engineering

King Fahd University of Petroleum and Minerals, Eastern Province , Dhahran 31261, Saudi Arabia

Abstract

Recently, web service became popular for Real-time Communication (RTC). It allows bi-directional, real-time communication between web clients and server. On the other hand, Data Distribution Service (DDS) middleware offers unified integration with high-performance due to its scalability, flexibility, real-time, mission-critical networks and rich QoS features. DDS is based on the publish/subscribe communication model. It improves RTC through its efficient and high-performance data delivery mechanism. This paper studies and investigates that how DDS is better for RTC. Experimental studies are conducted to compare text messaging using socket IO over DDS Web API. The result concerns the throughput satisfaction rate, round trip time and packet loss. In addition, we consider some of QoS of DDS during experimental work e.g. deadline, time based filter etc.

Keywords: Middleware, DDS, Mobile Communication, Round-trip Time, GPS, Wireless Communication ;

1. Introduction

Today communication has large impact on our daily life. Social communication to mission critical operation in every cases faster communication is an important factor. So we need to develop in this field of faster communication so that we can send and receive data quickly and reliably. Similarly, faster communication is high demand in transport system. Real time coordination with moving objects and display them in the map is a common challenge. Nowadays, there are many technology addressed this challenge. The web service is one of these. We used Socket IO[1] as a web service in this paper. Another way of faster communication is, DDS for RTC. It is an Object Management Group (OMG) machine-to machine middleware "m2m" standard that objects to permit dependable, real-time, high performance, scalable and interoperable data exchanges between publishers and subscribers. RTC permits you to exchange your content like multimedia, voice and text in real time. Text messaging means the transmitting and re-

* Basem Almadani. Tel.: +9663-860-7424 ; fax: +9663-860-3059.
** Abdullah Al Mamun. Tel.: +966-590-822924.
E-mail address: mbasem@kfupm.edu.sa, g201403680@kfupm.edu.sa

ceiving vehicle's current position in our experiment to evaluate performance in both socket IO and DDS. To this end, we implemented real-time platform using both socket IO and DDS WEB API at where publisher can publish samples data on a certain topic and connected subscriber can get updated data concurrently. We will use RTI Connext[2] to implement the publisher subscriber model in our experiment. This paper would like to evaluate the performance between socket IO and RTI WEB API in case of performance criteria and DDS QoS. There are many different technology used already to implement vehicle tracking[3 4 5] in real-time, but very few are successes in this field.

The system that desires data is called a service requester whereas the software that would process the demand and provide the data is called a service provider. Unlike outmoded client/server models, such as a web server/web page system, web services do not provide the user with a GUI. It instead share business logic, data and processes through a programmatic interface across a network is shown in fig 1(a). Web Socket is one of the perfect example of web service. It defines an API that enables web pages or mobile application to use its protocol for two way communication with a remote host. It can reduce network traffic latency compared to traditional HTTP polling and long polling techniques commonly used for full duplex connection by maintaining two connections. In addition, it is capable traversing firewalls and proxies. The web socket connection starts as an HTTP connection which then upgraded by replacing with the web socket connection over same underlying TCP/IP connection. Once connected, web socket data frames can be sent back and forth between the client and server in full duplex mode. Text and binary frame are supported as data packet. It is worth mentioned that all of configuration and specification of web service policies are standardized by World Wide Web Consortium(W3C).

Socket IO opens a dedicated port to communicate clients in both directions. When an event occurs on this port, all clients get noticed as well as updates instantly with a short delay. It allows user to publish and subscribe sample data in real-time communication. Socket IO is nothing but a javascript library that runs on a node js server, it has two parts: a client-side library that runs in a browser, others library runs on server. Both components have a nearly identical API. Usually node.js is an event-driven server scripting. Socket IO primarily uses the Web Socket protocol and it will automatically select the best suited real-time communication protocol at run-time per client. It is a set of stipulations standardized by the OMG. The DDS middleware is a known standard with fixed data-structures and attributes quantified by meta-information called topics[6]. Every topic defines a set of associated data samples with the same data-property and data-structure. For example, a topic named "speed" has to be stored into database after observing thousands of motor by using distributed set of sensors[7]. The publishers are entities that write and read the data-samples using a DDS-based middleware that is shown in fig 1(b). More precisely, a publisher consists of a set of data writer modules, each of which is used to write information on a particular topic. In contrary, a subscriber reads the data samples of topics by using its data reader modules. A topic is qualified by a wide set of Quality of Service (QoS) parameters that manage a number of aspects of the distribution of linked data samples. As an example, in order to safe web browsing, we may use content based filters to delete undesired data packets. Among several approaches for establishing communication between heterogeneous systems, publish/subscribe PS is a message oriented service where senders of messages is called publishers do not send messages to a specific receiver directly whereas receiver acts like a subscriber. In this case, published messages are classified into several topics with unique identifications. PS model is event driven in nature, so when an issue is generated publisher starts publishes data over the net to subscribers those express interest in one or more classes (topics) and only receive messages that are of interest that is shown in fig1(b). Due to its loosely coupled property, publish/subscribe architecture is more flexible and scalable for distributed systems; in our work, this architecture is represented by DDS standard.

Another approach is the traditional web service, which is already described above. The web service is a tightly coupled pattern where the programmer should specify the client/server's address and they have to work at the same time. That makes it less scalable than publish/subscribe pattern. In brief, Wang, N. et al.[8] summarizes the advantages of publish/subscribe over the client/server architecture as publish/subscribe is plug and play, loosely coupled, fault resilient, and inherently many to many architectures whereas client/server architecture is complex in development, tight coupling, fragile to a fault, and inherently one-to-one architecture. The main contribution of this paper is to examine the behavior of real-time vehicle tracking over both publish subscribe and client server architectures using the DDS middleware and web service API. Furthermore, we demonstrate the most important quality of service performance parameters of DDS and describe how these can be configured to improve transmission of sending and receiving samples over wireless networks. In addition, we compared the performance between web service and DDS. Experimental result will prove that DDS is very much applicable and a promising technology for vehicle tracking in real-time en-

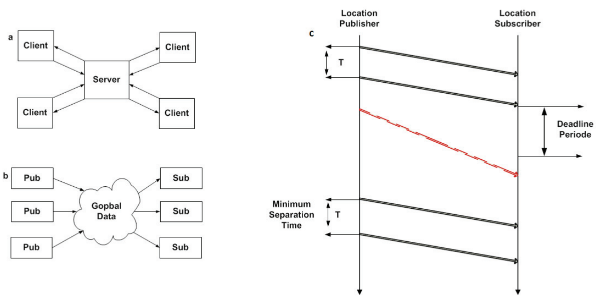


Fig. 1: Architectural view of (a) Client-Server (b) Publisher-Subscriber and (c)Time Diagram of Deadline and Time based filter QoS

vironment. There are some key features such as platform independent and reliability help it significantly improve the quality of tracking your transport system over heterogeneous platforms. The paper is organized as follows: section 2 gives a background on DDS QoS and describes how avoids DDS QoS confliction by configuring QoS in an advanced way. We demonstrate the experimental work and conduct a results analysis in Section 3. However, section 4 presents the previous work. Finally, conclusion is given in Section 5.

2. QoS Architecture for Vehicle Tracking

In this section, we review and discuss the QoS of DDS that can be adapted to improve vehicle tracking and reduce the effect of network congestion. Also, we provided an analysis on the use of GPS with Socket IO and DDS middleware in tracking application.

2.1. Supporting DDS QoS Polices

Many QoS policies are used by DDS middleware to support fast transmission of vehicle location over networks and also to minimize the latency. In this section, we investigate related QoS policies and showed how these can be used to support flawless and accurate location information of vehicles.

2.1.1. Deadline

Network congestion occurs when a link or node is overloaded and as a consequence, it results in packet loss, increased delays and blocking of connections at a time. A lot of research has been done for mitigating network congestion. In the middleware layer, a deadline QoS policy can be used for congestion detection and control, as illustrated in fig1(c) where red arrows shows the amount of time taken by sender to send a packet which is reached at destination after deadline.

2.1.2. Time-Based Filtering

The minimum separation time is the time gap between two successive packets that are going to be receive by the subscriber. This QoS policy used in tracking applications is to reduce application load (receiving rate) at the subscriber. For instance, the publisher is a server and subscribers are different devices that have different capabilities e.g. laptop, PDA, cell phones, or even sensors in WSNs, each one of these has to adapt the receiving rate based on its available resources using such policy. It is shown in fig 1(c) where T is indicating the minimum separation time. Note that the time-based filter value must be less than the value of deadline because the deadline is the maximum wait time for data update on the subscriber.

2.2. QoS Confliction

Meet the QoS satisfaction is a big challenge in case of any middleware due to confliction with each other. Although both deadline and time based filter QoS are important for vehicle tracking, a contradictory behavior exists between these two QoS. Choosing the value of QoS parameters of deadline depends on minimum separation time. We are

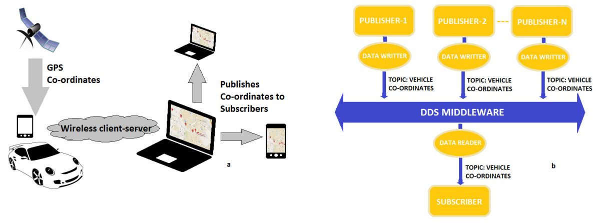


Fig. 2: Experimental setup for (a)Socket IO (b) DDS

proposing a solution here to handle unambiguous situations. If δ is Round-trip time per sample and ψ is number of samples then average Round-trip time (RTT) can be written as follows:

$$Avg.RTT = \frac{\sum_{i=1}^{\psi} \delta_i}{|\psi|} \tag{1}$$

In the case of Time Based Filter QoS, if Latency is ξ and minimum separation time is Δ then RTT can be written as follows:

$$RTT_{TBF} \geq 2\xi + \Delta \tag{2}$$

Algorithm 1 Proposed Solution

```
1: procedure DDS QoS
2:     $\tau \leftarrow$ Deadline
3:     if $\tau < (\xi + \Delta)$ then
4:         Packet Lost
5:     else
6:         Packet Receive
```

3. Experimental Work

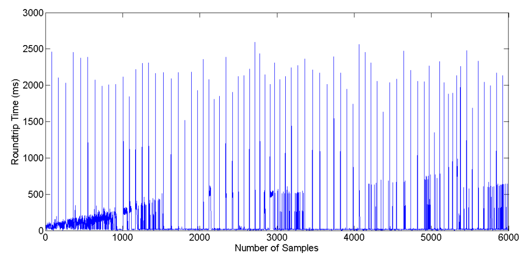
In this section, we experimentally evaluated the performance of Socket IO and DDS based vehicle Tracking and compare between these.

3.1. Hardware and Software Specification

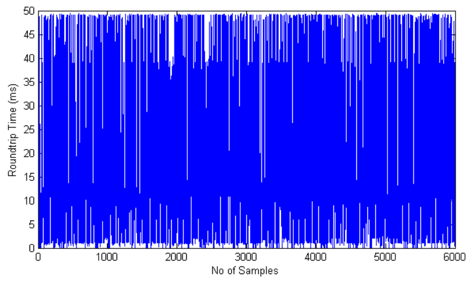
The experiment was carried using hardware and software tools; the measurement and monitoring tools and hardware platform specifications that are described in Table 1.

Table 1: Tools and Programs

Tool	Version	Use
Soket-IO Lib[1]	1.2.1	To Establish http client-server
Google Map JS Lib	3.0	Visualize vehicle position
Android SDK	JB 4.3	To publish co-ordinates
NodeJS	0.10.33	To server scripting
RTI Connext[2]	5.1.0	Real time vehicle tracking
Wireshark	1.2.3	Performance test



(a) Socket IO



(b) DDS

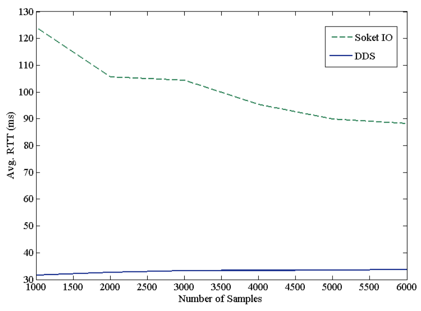
Fig. 3: The trend of round-trip time

3.2. Experimental Setup and Performance Metrics

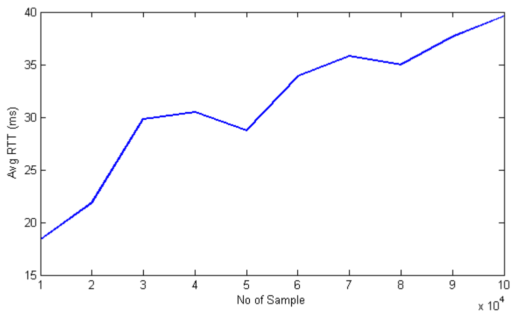
As shown in fig 2(a), this is architectural view of socket IO based vehicular tracker where smart phone is used to receive GPS coordinates from satellite and forward it to the node js server by using socket IO library and web service API. Actually a dedicated port is used in socket IO in a way that when a packet is arrived at this port, clients get a quick reply because clients keep listening always. All clients get samples (new co-ordinates) immediately when an event occurred at this port. This is the difference between request reply based client-server and event driven client-server. In this case smart phone/Tab can act both publisher and subscriber at the same time which is shown in fig 2(a). On the other hand, the DDS based test bed is shown in fig 2(b). In this case more than one publisher publishes samples directly on vehicle coordinates topic through data writer. Similarly, subscribers are receiving samples on this topic through data reader. The main architectural difference between client-server and DDS is publisher-subscriber communication. In DDS, publishers and subscribers are directly connected[9] and subscribers are getting samples from publishers directly whereas in client-server model client gets data from the server but server depends on other clients to get data samples[10]. Socket IO chat example[1] is used to make http server for observing vehicle position. In contrast, RTI Vehicle Tracking system[2] is used to implement vehicle Tracking over DDS. Initially we send x,y coordinates that makes sample size 28 bytes in both socket IO and DDS technology. The protocol RTPS2 is used in DDS and UDP in Socket IO. The QoS parameters are adjusted to meet the exist in architectural network link specification; for example, the deadline is adjusted infinite time whereas minimum separation time is 1ms. These parameters are suitable for dedicated and fast networks such as WiFi.

Table 2: RTT Comparison between Socket-IO and DDS

Type	No. of Samples	Avg. RTT.*(ms)*	STD.*(ms)*	Min.*(ms)*	Max.*(ms)*
Soket-IO	1000	124.12	220.02	6	2458
	2000	105.61	234.91	6	2458
	3000	104.41	255.31	6	2590
	4000	95.36	255.82	5	2590
	5000	89.89	258.25	5	2590
	6000	88.28	262.22	5	2590
DDS	1000	31.61	14.23	0.03	49.35
	2000	32.64	14.40	0.03	49.35
	3000	33.33	14.62	0.03	49.35
	4000	33.40	14.59	0.03	49.35
	5000	33.54	14.62	0.03	49.35
	6000	33.75	14.65	0.03	49.35



(a) Soket IO and DDS

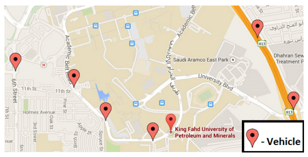


(b) DDS time base filter QoS

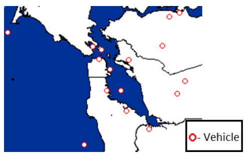
Fig. 4: Average round-trip time

3.3. Result and Analysis

To compare performance between Socket IO and DDS, initially we consider one to one wireless communication with 25Mbps bandwidth connection, maximum number of samples is 6000. Moreover, experimental data is collected by executing same experiment repeatedly that is 31 times. However, the trend of round trip time of socket IO and DDS is shown as a line chart, respectively in fig 3(a) and fig 3(b) where the x-axis is representing numbers of samples and y-axis is indicating the round trip time in millisecond (ms). According to graph 3(a), a common phenomenon is observed that RTT goes to pick in average 100 samples. This spikes are found in the timeline chart in socket IO because of clearing cache. However, It is clearly observed that the RTT of DDS is more stable than the trend of Socket IO. Moreover, it shows that almost same RTT is carried over the total experiment in DDS. The round-trip time of socket IO and DDS is shown in table 2 where average round-trip time (Avg. RTT), Standard deviation (STD), Minimum RTT and maximum RTT on different number of samples are recorded for both socket IO and DDS middleware. In case of Socket IO the maximum RTT is 2590ms for the number of samples is 6000 whereas it is only 49.35ms for DDS. Similarly, it shows that minimum RTT of DDS is too small than RTT of Socket IO that is a big indication that DDS is maintained much real-time behavior than Socket IO. Moreover, standard deviation of RTT in DDS is almost similar in every case. Although both socket IO and DDS has recorded large round-trip time, but the percentage of samples having less than Avg RTT are more than 95% of total samples which is good obviously. The average round-trip time comparison graph is shown in fig 4a. As above graph, x-axis represents the number of samples and y-axis indicates Avg. RTT. The Avg. RTT of socket IO is high for low samples and it is slowly decreasing related to the increase of sample number. In contrary, the Avg. RTT of DDS is almost similar for every size of sample number. It is clearly observed that the average RTT of DDS is much smaller than Web service for any sample number. Average RTT is calculated according to the formula (1). Finally it is a transparently indication that DDS is more real-time and efficient technology than a web service. The position of all vehicles is shown graphically for both Socket IO and DDS respectively in fig 5a and fig 5b. Basically from the comparison of DDS with Web service, we learned, DDS is better than web service.



(a) Google Map



(b) DDS Map

Fig. 5: Vehicle position

An Average RTT of Time Based Filter QoS is shown in fig 4b where x-axis indicates number of samples and y-axis represents Avg. RTT. Here 2ms is set as a minimum separation time to test Time Based Filter QoS and we got a good result with no packet loss and better round trip time. In this case Avg. RTT is calculated by using formula (2). We use 120000 samples to test both scalability and Time Based Filter QoS. However, in every case DDS comes up with a good performance that motivated us to use real time vehicle tracking for faster communication. The Deadline Qos is tested according to the algorithm 1. It solved confliction between two QoS respectively Time based filter and deadline in a way that when the deadline is less than the sum of latency and minimum separation time, packet will no longer received by the subscriber. Otherwise, it received the packet always. We found zero packet loss in this test because the deadline is 1 min. and all packets are arrived at subscriber before deadline.

4. Related Works

Although much research has been done on vehicle tracking, but very few of these are based on middleware. Similarly, much research has been done in Publish/Subscribe (Pub/Sub)[11,12,13,14,15], but very few among these supports big-scale mobile networks and simultaneously it ensures the QoS for the field of cellular communications, especially the delivery of the low latency message and aforesaid reliability[16,17,18].

A DDS based middleware called Scalable Data Distribution Layer (SDDL)[19] presents the OMG DDS standard based communication service middleware that supports unicast, groupcast and broadcast and live tracks with more than thousand cellular nodes.Two protocols are used in this middleware namely RTPS wire protocol for communication among the stationary nodes and mobile reliable UDP protocol for communication among the mobile nodes. The major contributions of SDDL are (i) It is capable to optimized extension of live communication with several mobile nodes with out own DDS supports by using highly optimized UDP protocol which is IP address independent; and (ii) an extensible and adaptive communication middleware which supports mobile node handovers and broadcast, group cast etc, where the logic of group defining is subjective. David, L. et al.[20] presented same things in large scale integration for thousand of nodes. They also used SDDL for developing an application for mobile device to support vehicle inspection by traffic police. Traffic surveillance[21] describes the problems related to tracking based on feature which is basically a real time system that is nothing but a prototype, and shown system performance using large data. In this paper, they focused on segmentation of vehicle and inspection, also they collected tracking data as computation of traffic parameters. For large-scale mobile system, a middleware called Scalable context-Aware Middleware for mobile environments (SALES)[22] is developed. Two main terminologies used are: Quality of Context (QoC) and Context Data Distribution Level Agreement (CDDLA) in this paper. QoC is associated with context information distributive service whereas CDDLA is a quality agreement between consumer and producer imposed by the middleware.The architecture[23] middleware supports mobile nodes and provides reliable data delivery. It also supports handover by switching the wireless access points. The mobile nodes in this middleware execute light version of the DDS whereas the fixed nodes execute the full version of the DDS. In case of Industrial scenarios[24] proposes a real time Automatic Vehicle Location (AVL) and Monitoring system for pilgrims road transport coming towards the city of Makkah in Saudi Arabia based on DDS. They proposed a huge system based on the DDS standard of middleware considering about 50000 buses as mobile nodes.They proposed an automated solution based on Real-time Publish-Subscribe middleware for real time tracking and monitoring of vehicles as well as pilgrims. DDS is used as middleware because of its Data Centric and Asynchronous communication model along with a rich set of QoS policies. The CAN BUS[25] of DDS over CAN simulator that interacts with the main factors presented by the DDS spec. Their goal is optimized network behavior and improve the consistent data delivery by integrating the DDS QoS parameters.

5. Conclusion

This paper introduced a performance evaluation for both web service and DDS in case of vehicle tracking in real time over wireless medium. From empirical results, it can be concluded that this technology is a promising technology for distributed moving object over networks. Since it has low latency, and causes lesser packet loss, it gives more control on vehicular tracking through the use of a rich set of QoS polices that are provided by the DDS middleware. Clearly observed that DDS performance is much better than the performance of web service. We want to continue our research for large scale and we have planned to implement fleet control for Hajj pilgrims in state Makkah as future work. We will be able to publish more topic e.g. speed and vehicle information in near future as well as we will integrate this technology into animal tracking for national forest department.

Acknowledgements

Department of Computer Engineering, KFUPM. Real Time Innovations(RTI). Google Map is a trademark of Google Inc. Socket IO is open source and run by contributors.

References

1. (MIT), S.I.O.S., CONTRIBUTORS, R.B.. Socket-io documentation. http://socket.io/docs/; 2014, .
2. http://www.rti.com/company/, . Rti connext. https://community.rti.com/documentation/rti-connext-dds-510/; 2014, .
3. Kotte, S., Yanamadala, H.B.. Advanced vehicle tracking system on google earth using gps and gsm ????;.
4. Verma, P., Bhatia, J.. Design and development of gps-gsm based tracking system with google map based monitoring. *International Journal of Computer Science, Engineering and Applications (IJCSEA)* 2013;**3**(3):33–40.
5. Santoso, F., Malaney, R.. Tracking-based wireless intrusion detection for vehicular networks. In: *Vehicular Technology Conference (VTC Fall), 2011 IEEE*. IEEE; 2011, p. 1–5.
6. Al-Madani, B., Al-Roubaiey, A., Baig, Z.A.. Real-time qos-aware video streaming: A comparative and experimental study. *Advances in Multimedia* 2014;**2014**.
7. Detti, A., Loreti, P., Blefari-Melazzi, N., Fedi, F.. Streaming h. 264 scalable video over data distribution service in a wireless environment. In: *World of Wireless Mobile and Multimedia Networks (WoWMoM), 2010 IEEE International Symposium on a*. IEEE; 2010, p. 1–3.
8. Wang, N., Schmidt, D.C., van't Hag, H., Corsaro, A.. Toward an adaptive data distribution service for dynamic large-scale network-centric operation and warfare (ncow) systems. In: *Military Communications Conference, 2008. MILCOM 2008. IEEE*. IEEE; 2008, p. 1–7.
9. OMG, . Web-enabled dds. http://www.omg.org/spec/DDS-WEB; 2013, .
10. Tanenbaum, A.. *Computer Networks chapter six*. Prentice Hall Professional Technical Reference; 4th ed.; 2002. ISBN 0130661023.
11. Huang, Y., Garcia-Molina, H.. Publish/subscribe in a mobile environment. *Wireless Networks* 2004;**10**(6):643–652.
12. Nogueira, J., de Middleware, T.. A large-scale and decentralised application-level multicast infrastructure ????;.
13. Carzaniga, A., Rosenblum, D.S., Wolf, A.L.. Design and evaluation of a wide-area event notification service. *ACM Transactions on Computer Systems (TOCS)* 2001;**19**(3):332–383.
14. Terpstra, W.W., Behnel, S., Fiege, L., Zeidler, A., Buchmann, A.P.. A peer-to-peer approach to content-based publish/subscribe. In: *Proceedings of the 2nd international workshop on Distributed event-based systems*. ACM; 2003, p. 1–8.
15. Pietzuch, P.R., Bacon, J.M.. Hermes: A distributed event-based middleware architecture. In: *Distributed Computing Systems Workshops, 2002. Proceedings. 22nd International Conference on*. IEEE; 2002, p. 611–618.
16. Mahambre, S.P., Kumar, M., Bellur, U.. A taxonomy of qos-aware, adaptive event-dissemination middleware. *Internet Computing, IEEE* 2007;**11**(4):35–44.
17. Corsaro, A., Querzoni, L., Scipioni, S., Piergiovanni, S.T., Virgillito, A.. Quality of service in publish/subscribe middleware. *Global Data Management* 2006;**19**:20.
18. Esposito, C., Cotroneo, D., Gokhale, A.. Reliable publish/subscribe middleware for time-sensitive internet-scale applications. In: *Proceedings of the Third ACM International Conference on Distributed Event-Based Systems*. ACM; 2009, p. 16.
19. David, L., Vasconcelos, R., Alves, L., André, R., Baptista, G., Endler, M.. A large-scale communication middleware for fleet tracking and management. In: *Simposio Brasileiro de Redes de Computadores e Sistemas Distribuidos (SBRC 2012), Salao de Ferramentas*. 2012, p. 964–971.
20. David, L., Vasconcelos, R., Alves, L., André, R., Endler, M.. A dds-based middleware for scalable tracking, communication and collaboration of mobile nodes. *Journal of Internet Services and Applications* 2013;**4**(1):1–15.
21. Coifman, B., Beymer, D., McLauchlan, P., Malik, J.. A real-time computer vision system for vehicle tracking and traffic surveillance. *Transportation Research Part C: Emerging Technologies* 1998;**6**(4):271–288.
22. Corradi, A., Fanelli, M., Foschini, L.. Adaptive context data distribution with guaranteed quality for mobile environments. In: *Wireless Pervasive Computing (ISWPC), 2010 5th IEEE International Symposium on*. IEEE; 2010, p. 373–380.
23. Herms, A., Schulze, M., Kaiser, J., Nett, E.. Exploiting publish/subscribe communication in wireless mesh networks for industrial scenarios. In: *Emerging Technologies and Factory Automation, 2008. ETFA 2008. IEEE International Conference on*. IEEE; 2008, p. 648–655.
24. Almadani, B., Khan, S., Sheltami, T.R., Shakshuki, E.M., Musaddiq, M., Saeed, B.. Automatic vehicle location and monitoring system based on data distribution service. *Procedia Computer Science* 2014;**37**:127–134.
25. Rekik, R., Hasnaoui, S.. Application of a can bus transport for dds middleware. In: *Applications of Digital Information and Web Technologies, 2009. ICADIWT'09. Second International Conference on the*. IEEE; 2009, p. 766–771.