Journal of Ubiquitous Systems & Pervasive Networks
Volume 7, No. 1 (2016) pp. 19-27

Content-based Filter Publish Subscribe Model for Real-time WSN applications

Mohammed Mahyoub[*a], Anas Al-Roubaiey[b], Gamil Ahmed[c]

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

{G201405280[a], roubaiey[b], 201302310[c] @ kfupm.edu.sa }

Abstract

In the recent years, the publish/subscribe (pub/sub) communication model has emerged as a suitable communication paradigm for large-scale distributed systems. That is due to its effective decoupling properties for the network's participants in time, space, and synchronization. These properties are well-suited for Wireless Sensor/Actuator Networks (WSAN) applications. Data Distribution Service (DDS) is a well-known standard in the academic and industrial communities for supporting real-time distributed systems based on the pub/sub model. TinyDDS is a light weight and partial porting of DDS middleware to WSN platforms. The main objective of this paper is to use TinyDDS standard-based solution to minimize the energy consumption and maximize throughput of WSANs when applying the pub/sub interaction scheme, while maintaining the content-based filter QoS support. Adding content-based filter to the default TinyDDS (DTDDS) enable the WSAN to gain high performance in terms of packet delivery ratio and reduce the power consumption and we called this addition as CFTDDS. The Experiments s conducted in this work prove the efficiency of our proposal CFTDDS.

Keywords: *Content-based; WSN; publish/subscribe; Middleware.*

1. Introduction

Sensor networks are composed of tens/hundreds low-priced and tiny devices with limited capabilities that are deployed to an area of interest to monitor the behavior of a particular phenomenon. In conventional applications of single base station/sink WSN, the movement of data flow from the sensors to the monitoring application is usually through a sink node. In which all deployed sensors gather and transmit the data to the sink node using pattern of one-to-many communication [1]

Therefore, the core task on WSN was to sense and gather the data from the neighboring area with no action done. Several applications benefit from this functionality, such as environmental monitoring, Structural Health Monitoring (SHM), Human Health Monitoring (HHM), habitat monitoring, and military surveillance. However, Because of the recent developments in the technology of sensor-based network the Wireless Sensor and Actor Networks (WSAN) have been introduced as enabling technology for in-network decision making, where the network can sense and respond with no need to go to external and control applications [2].

[*] Corresponding author.
E-mail: g201405280@kfupm.edu.sa

DOI: 10.5383/JUSPN.07.01.004

After knowing what WSAN is, we define the publish/subscribe model and its suitability for WSAN. A publish/subscribe (pub/sub) paradigm is a messaging based communication model, where senders, called publishers, send their data to a logical data space, called middleware, without knowledge of who or where are the receivers, called subscribers. Similarly, subscribers receive only the data of interest, without knowledge of who or where are the publishers. [3]

The Pub/Sub interaction paradigm is designed to suite large-scale distributed real-time applications. Oh et al. [4] have done an appropriateness analysis for pub/sub scheme, their main remarks were as follows:

- The model of Pub/sub has advantage when system is large and many clients have shared a transferred data; which is usually the case in the networks of sensor in which enormous number of sensors are deployed to deliver the information of monitored object to several sinks and/or actuators.
- Pub/sub model is appropriate when updating of data or events doesn't occur frequently. For example, event-based applications that mainly monitor and control distributed systems (e.g. WSAN).
- Pub/sub model is appropriate when the degree of common interest is high. For example, in WSAN applications the data collected by sensors highly has a common interest by the multiple sinks, applications, or actuators.
- Pub/sub model is appropriate in less user intervention applications more than request/replay model.
- In pub/sub model the updates of data are directly provided to subscribers which is more appropriate for real-time applications where deadline is strict or short. For example, in battlefield surveillance WSN.
- When clients rarely use published data, a model of pub/sub is unsuitable.

The robustness and scalability of the paradigm came from its separating features in space, synchronization, and time [5]. In particular, these features make it more appropriate for applications that use data-centric sensor network. Furthermore, the applications of sensor network have definite properties making Pub/Sub middleware the suitable solution for such environments [6][7] [8].

Different ways for disseminating the aggregated data or the event values depending on the subscribers' interest (i.e., receiving only the events they are interested) in and this resulting in variants subscription models used with the pub/sub in WSN systems. In the art of literature, three subscription models are appeared: topic-based model, content-based model and type-based model.

In topic-based model, what the publisher interested in will be declared as a particular topic and all the updated samples of that topic will be received by the subscription. So, no filtering will be applied to the subscribed topic and in a simple word the subscription is a topic's specification. All early sub/sub model, this type is adopted as a solution for the subscription mechanism. The main drawback of the topic-based model is the limitation which the subscriber faces when it needs to express its interests and this make the subscriber not able to declare to subset values of a specific topic [9].

Let's take the heat monitoring system as an example to illustrate the drawback related to this type of subscription, In this system, the sensors are considered as publishers which periodically sense the temperature degree (i.e. the topic) of the surround environment and publish these values in the network, and the actuators are considered as subscribers which acts as a controller for the alarms or cooling system.

The actuator interested to subscribe not to all the values related to the temperature topic published by sensors but they interested on values when the temperature degree is greater than some threshold 30o in case of activating the alarm or interested to receive the topic when temperature degree goes higher 50^{o} in case of activating the cooling valve . So, subscribers receives different patterns from the published topic based on their predefined interest.

In the content-based model, subscribers express their interest by defining some constraints over the content of published data from the topic they want to receive. This type of subscription more appropriate in WSN.

Although the publishing of the updated data samples can be done through the publisher-to-subscriber principle, filtering the published data at the subscriber end is not practical for some application (Alarm system, since it interest in some condition constraints of the events but not all updated events). So, disseminating all the updated samples is high expensive in term of high overhead consumed the network bandwidth resulting in high delay and latency and causing in decreasing the system's throughput.

In this paper, TinyDDS [10] is middleware is selected because it is standard based middleware. It is the tiny (Lightweight) version of Data Distribution Service (DDS) standard that is standardized by the Object Management Group (OMG) organization in 2003 [8]. Recently, DDS has received a lot of attention from research community and industry. TinyDDS is intended for sensor network to allow them to be seamlessly integrated into enterprise networks, and to facilitate the support of Quality of Services (QoSs).[11]

However, TinyDDS still needs a lot of contribution to become stable and recommended as the middleware standard for sensor networks. In this paper a content-based filter QoS will be integrated with the TinyDDS to improve the performance and efficiency of the WSN in

term in increasing the throughput and decreasing the delay to utilize the WSN in more efficiency manner. Moreover, we perform an intensive simulation study to evaluate the actual performance of adding content-based filter QoS to TinyDDS.

2. CFWSN Architecture and Communication Model

2.1. CFWSN Architecture

In this section, the CFWSN Architecture and algorithm will be explained and discussed. Figure 1 shows the architecture of CFTDDS. Basically CFTDDS is TinyDDS middleware with proposed QoS added to it which is content-based filter. As mentioned in the previous section, TinyDDS uses DDS standard. According to publish/subcribe model implemented by TinyDDS, this model includes four main entities publisher, subscriber, pub/sub service, and the Application Programming Interfaces (APIs).

DDS associates with every topic in the network two main components: Data Writer (DR), at the publisher side, and Data Reader (DR), at the subscriber side. The CFTDDS basic mechanism is implemented in the DW and DR, therefore, after the modification these components are referred to as CF-DW, and CF-DR, where R stands for Reliable.

On CF-DW, the checking, filtering, and classifier mechanisms are implemented, whereas the content-based subscription conditions is implemented on CF-DR. As shown in the architecture, the CFTDDS middleware intermediates between the application and the platform details, such as TinyOS [9] protocols and Sensor/Actuator hardware. Thereby, the application can interact with the system only through the DDS API interfaces, which makes the application development easier.

The middleware used in this paper is TinyDDS. Although there are many middleware in the literature but the following reasons are the main motivations to use TinyDDS specifically.

TinyDDS is an open source based on well-known DDS standard real time middleware and as a result, this middleware provides the interoperability or the WSN to integrate with the enterprise networks. Also, many of the Quality of Services (QoS) are supported by TinyDDS and applies some constraints on these QoS to suite the resource lacking on WSN platforms.

In addition, the underlying layer complicated details are hidden from the application layer so the developers implement a single application for different platforms. For example, on application can be built for different WSN platform such as telosb [12], mica2 [13], iris [14], and micaz [15]. So, TinyDDS supports the portability in term of ability to integrate multiple different WSN platforms.

Finally, important feature advantages TinyDDS inherited from the publish/subscribe communication model is supporting the scalability due to that this paradigm decouple the system in space, time, and synchronization as mentioned in the introduction section.

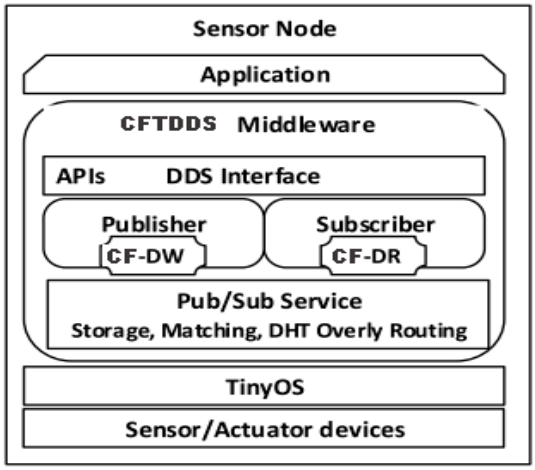


Figure 1: CFTDDS Architecture

TinyDDS uses Broker-based subscription model where a specific node called Rendezvous node is selected depending on DHT algorithm for each topic to act as a central broker for handling the communication between the publishers and subscribers ends. But the centralized approach is not suitable in the WSN environment due to the constraints of limit resources of the sensors and generally this will lead to a problems as single point of failure and bottleneck. As result, this will rapidly exhaust the node energy, and eventually ends the network life time while the network still has adequate residual energy.

In this work, the Hybrid TinyDDS (HyTDDS) communication model proposed in PhD thesis done by [14] is used, two phases are needed for each participant (i.e. Sensor or Actuator) in the pub/sub WSN system: (1) Discovery phase and (2) Data Dissemination phase. In this method, each publisher will take care and maintain its interested subscriber list (subscriber data base) and they get their list from the Rendvous node (RN) which match the subscriber's interest and publisher's advertisement to create a subscription list and send it sent to the matched publisher. . So, the RN will works only in the discovery phase, to forward the subscription messages to the matched publishers [16].

The reasons behind using HyTDDS comes from the nature of the WSN applications where multiple subscribers (i.e. Actuators) are communicating directly with the sensors to get their readings). So to eliminate the bottleneck and single point of failure problem caused by centralized Rendvous node in data dissemination phase, the HyTDDS approach is more appropriate in WSN environment.

2.2. CFTDDS Algorithm

In CFWSN, the subscribers express their interest using particular constraints using comparison operators (e.g. =, <,>, >=, <=). Note that some of the subscribers may interest in all sensed value of some topics such as subscribers acts as a historical database so in this case no constraints will be applied on the subscription interest.

In the publisher side, the algorithm 1 shown below (i.e. simple pseudo code) is executed where the publishers (i.e. sensors) sense their around environment to get some reading values (e.g. the temperature, pressure, or humidity degree). Sensors check their subscription list created in discovery phase mentioned in the previous section (line 2). If the read value matches one or more subscription constrain in the subscription list then the read value is published to the matched list (line 10), otherwise the sensed value is ignored (line 8).

For more clarification for the proposed CFTDDS algorithm, figure shows the flowchart describing the sequence events executed in the publisher side. Applying CFTDDS algorithm will enable the publishers to publish their reading not always but depend on the constraints applied to the subscriber's subscriber's interests, so, many advantages will be gained such as less traffic in the network resulting in less congestion and packets dropping, increasing the throughput and decreasing the end to end delay and power consumption.

Algorithm 1: CFTDDS Pseudocode

Parameters.: *Subscription_list, satisfied_List, Sensed_value*

CFWSN Procedure

1 *satisfied_List* □ *Empty*

2 **foreach** *j* ∈ *Subscription_list do*

3 **if** *Sensed_value* ∩ *Subscription_list (j).constrain != empty* **then**

4 *satisfied_List* □ *Subscription_list (j).subscriber;;*

5 **end if**

6 **end for**

7 **if** *satisfied_List == empty* **then**

8 **Ignore** *Sensed_value;*

9 ***else***

10 **publish** *Sensed_value* to *satisfied_List;*

11 **end if**

12 end procedure

3. Performance Evaluation

3.1. Performance Metrics

To evaluate the performance of this paper, three performance metrics are used which are Packet Delivery Ratio (PDR), end-to-end delay and energy consumption. This subsection gives a definition for these metrics.

1. Packet Delivery Ratio (PDR):

Is defined as the division ration between total successful packets received by all publishers and the total of packets sent by all publishers. The larger the packets send to the network, the larger the congestion and buffer overflow are occurred. As a result, high rate of packets dropping is expected and so the PDR will be decreased.

2. End-to-end delay:

Is defined as the total delay for all successful received packets by the subscribers. This delay includes transmission delay, queuing delay. It is expected that when the traffic load goes high then the queuing delay also goes high, as a result, end-to-end delay will be increased. Although this metric highly depends on the underlying protocols, the evaluation of two framework CFTDDS and DTDDS will be conducted using the same underlying protocols to get more accurate results and fair comparison.

3. Energy consumption:

It is assumed basically that the power source of the sensors to work is batteries and it is difficult to recharge these batteries due to large number of sensors in WSN. For this reason, the power consumption is the critical issue in WSN, so, this type of networks require that the communication and all processes and systems work within it minimize the power consumption order to maximize the life time of WSN.

In this evaluation, the total power consumption is calculated for whole network by accumulate the power of all sensors and this summation is divided by the total initial power for all sensors to give a percentage ratio.

4. Memory footprint:

Due to the limit resources challenge the wireless sensors devices, so the memory consumption is acritical metric in evaluating sensor applications and protocols. Two measures are applied to evaluate memory consumption which are Random Access Memory (RAM) and Read Only Memory (ROM) memory footprints with term of the byte are needed to enable the sensor handle the specific application properly.
In this work, after running the DTDDS and CFTDDS, the number bytes in of both RAM and ROM are measured. This evaluation is done on different platforms namely: mica2, micaz, and iris.

3.2. Simulation Setup

3.2.1. Data Rate

To evaluate the performance of our content-based proposed solution, three different data rate scenarios are used to test the efficiency of this work. These scenarios are called slow and normal and fast data rate. Firstly, we mean with the data rate is the number of sensing times of packets published by the sensors (i.e. publishers) towards the subscribers. In this section the reasons behind this variation is justified. Some wireless sensors application

In this evaluation the data rate varies by changing the Inter Packet Interval (IPI) for the three proposed types of data rate SDR, NDR and FDR where the IPI are 1, 0.25, and 0.1 seconds (i.e. 1, 8, and 16 packets/second) respectively. The IPI values are obtained from different simulation experiments to get stable and accurate results.

3.2.2. Network Topology

In this evaluation, the topology of the network used is illustrated in Figure 2. Note, this topology is considered as one of the experimental scenarios done in this work, since in this explanation just for clarify the used topology. All scenarios in this evaluation have a square grid topology composed of 49 nodes is deployed in a 100x100 square meter area with a single subscribers and different number of publishers for each scenario (i.e. 2, 4, 8 and 12 publishers).

Here, the node with id 48 at the upper right corner positon is considered as the base station/subscriber, and the publishers are distributed randomly within the tested square grid. For example in the first scenario, two nodes at the bottom left corner with ids 7 and 1 are the senders/publishers, the remaining nodes are relay nodes. Thereby, the maximum number of hops nearly 10 hops, sometimes due to network congestions/failures the routing protocol selects longer paths.

Figure (2) shows the distribution of the publishers in the tested squared area (the blue colored cells represent the publishers since there is only on subscriber is repented by green colored cell positioned on the top right corner). The cells with the white color represent the relay node used for packets forwarding purpose. It is worthy to note that the number labeled in the cell represents the ID of that sensor. This figure shows all scenarios used in our evaluation where 2, 4, 8, 12 publishers are used.

Table 1 shows the simulation parameters setup.

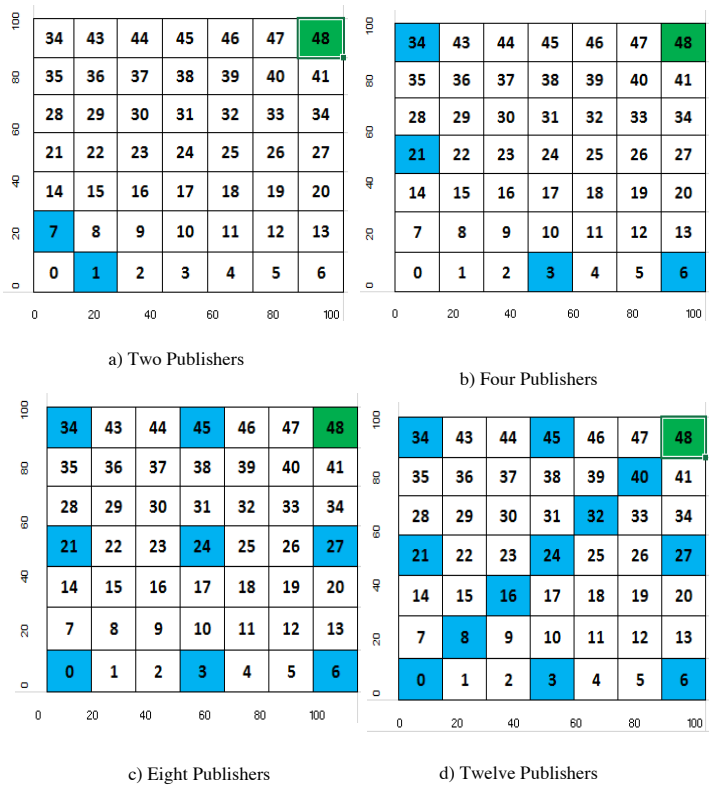


Figure 2: Publishers and subscriber distribution in the network topology with area size of 100*100m

Figure 3: Energy consumption using CFTDDS and DTDDS

Table 1: simulation parameters setup

Parameter	Value
Topology	Squared grid
Area	100 X 100 Meter^2
Number of Nodes	49
Simulation time	500 seconds
Radio model	Chipcon CC2420 [17]
Mote platform	micaZ
Data rates	SDR, NDR and FDR
Number of publishers	2, 4, 8, and 12
Message size	20 bytes
Maximum hops	10
Runs per results' data point	10

3.2.3. Tested Application

The application implemented to evaluate the proposed solution CFTDDS is basically acts as a collector for some reading values from the around environment such as temperature, pressure, or humidity. This application will work on two scenarios. The first scenario will use the default TinyDDS without content-based filter QoS and this scenario is called as DTDDS in our evaluation. The other scenario is using the CFTDDS where the content-based filter QoS is added to the default TinyDDS.

3.3. Results and Analysis

In this work, TinyOS SIMulator (TOSSIM) (Levis, Lee, Welsh, & Culler, 2003) is used to evaluate the performance of CFWSN. The performance evaluation of CFWSN is conducted depending on the three metrics mentioned above which are power consumption, PDR, and end-to-end delay when the number of publishers changed (i.e. 2, 4, 8, and 8 publishers) and the data rate varies (i.e. SDR, NDR, and FDR). The results gotten from using proposed CFTDDS will be compared with result gotten using the Default TinyDDS (DTDDS).

Firstly, the power consumption of the whole publishers participate in the proposed WSN will be calculated. In this evaluation, the average power needed for sending a packets all runtime expected to be low when the content-based filter As a result, the power consumption will be decreased comparing if all sensed data are published without any filtering in DTDDS case.

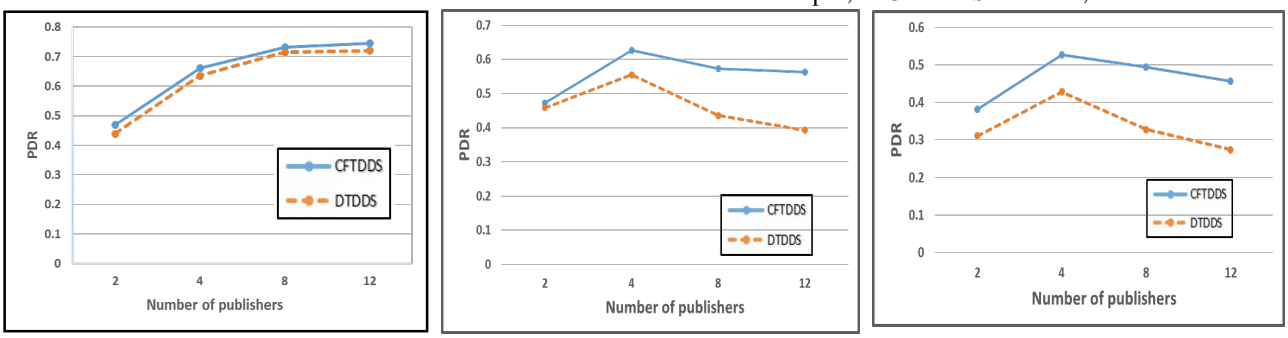
The Plotting of the power consumption results is shown in the figures (3) when different number of publishers are used and one packet per second data rate added to TinyDDS comparing with Default TinyDDS. Sensors in CFTDDS will send the packets to the subscribers only when the condition of the subscriber is satisfied. So the number of published data is decreased.

The energy consumption percentage is calculated as a division of the total consumption and the initial energy of the whole network. The initial energy of each node in the network was 2000 mAh, which is equivalent to 21600 Joules. It is clear that the proposed CFTDDS gives the minimum power consumption in all different publishers' number and outperform the DTDDS.

Secondly, Figure 4 (a, b, and c) shows the effect of the network load on the PDR on DTDDS and CFTDDS. The PDR tests the network reliability. In our test, we use the reliability QoS of TinyDDS for the purpose of fair comparison. It should be noted that in this study our main concern is to evaluate the middleware overhead with and without using content-based filter QoS.

In the DTDDS and CFTDDS scenarios, the PDR varies very little with the SDR (one packet per second), which means that the overall network load of the network is low in the two scenarios. In contrast, the middleware scenario has larger variation with the increase of data rate (NDR and FDR, i.e. 8 and 16 packets per second, respectively), because DTDDS has more control traffic due to sending many packets without any filtering used in publisher side. The CFTDDS overhead can be extracted from the drop of the network performance, also represented by PDR value, where it is nearly 10% less compared to the baseline scenario.

For example, in CFTDDS scenario, when FDR is used



a) PDR with 1 packet /sec data **b)** PDR with 8 packet /sec data **c)** PDR with 16 packet /sec data

Figure 4: Packet delivery ratio comparison

(i.e. data rate equals to 16 packets per second), the mean value that is calculated from 10 runs is 0.45 compared with 0.31 in case of DTDDS. From the PDR in Figure 4.c, we can see that the network in case of CFTDDS scenario is less congested than in DTDDS scenario. As a result, the average packet end-to-end delay is higher in case of the DTDDS scenario as shown in Figure 5.

The difference in the delay between both scenarios depends on the network traffic load, where the difference is not that much because in CFTDDS the time needed to computation (searching to match the subscription constraints) will be added to the delay applied to the packet when it is sent to the subscriber. This computation time overhead make the difference between the delays in both scenarios is nearly the same.

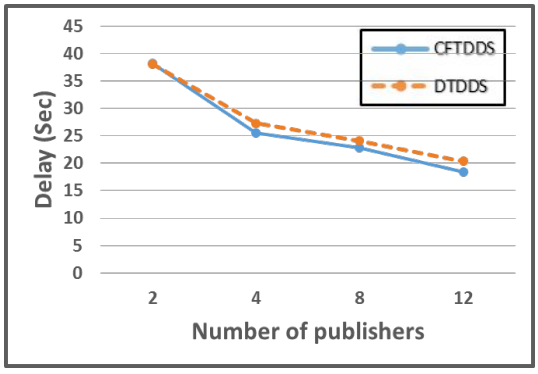


Figure 5: End-to- end delay comparison

The difference in delay goes higher when the data rate and the number of publisher increase. That is because when the network is not overloaded, the packet delay almost the same when we have the same packet size. Thus, the figure shows that the difference in the delay nearly ranging from 40 *ms* (in case of two publishers) to 200 *ms* (in case of 12 publishers). That is because the network in case of CFTDDS scenario has lightweight load even when using different number of publishers the packets reaches the subscriber using almost same number of hops. Whereas, in the other scenario the network was overloaded which results in more queuing delay and might be more hops due to network congestion.

Intuitively, the delay decreases as network load decreases and vice versa. However, in case of our evaluation, the network topology (distribution of the publishers in the network) make an opposite thing. That is because when the number of the publisher increase, they are distributed near to the subscriber position. So the average delay will be decreased until some saturation point and after this point may collapse occurs which causes suddenly increasing in the delay because the congestion in the network.

Finally, the memory requirements in each scenario are illustrated in Table 2. This figure describes the ROM and RAM consumption for three platforms: iris, mica2, and micaz. This table includes the exact number of bytes needed by each scenario. For example, for the iris platform, the DTDDS scenario uses 20290 bytes in program flash memory (ROM) and 5889 in RAM; whereas, the CFTDDS scenario allocates 24344 *bytes* in ROM and 6970 *bytes* in RAM for the same iris platform.

Thereby, we can evaluate the memory overhead of a sensor device when a content-based filter QoS is added. In iris platform, the CFTDDS overhead versus the DTDDS middleware application is about 1.2% more memory space in ROM and also 1.2% more memory space in RAM. This is considered a small difference in the memory space, relative to good throughput and saved power consumption gained from adding the content-based filter QoS. However, from iris datasheet these values are still acceptable where it has 48 KBytes ROM, and 10 *KBytes* RAM, and also one *MBytes* for logs, measurements readings, etc.

Table 2: Memory footprint comparison

		RAM	ROM
IRIS	DTDDS	5889	20290
	CFTDDS	6970	24344

		RAM	ROM
Mica2	DTDDS	5569	18616
	CFTDDS	6650	22624

		RAM	ROM
Micaz	DTDDS	6095	21878
	CFTDDS	7176	25932

4. Conclusion

In this work, content-based filter is added to TinyDDS standard-based middleware called CFTDDS and from the obtained result it is clear that our proposal prove its efficiency in terms of throughput and reducing the power consumption. We intend as a future work to do the empirical experiments instead of simulation to get an accurate results. The empirical experiments will be conducted using TelosB motes available in our DDS lab.

Acknowledgments

The authors would like to thanks Computer Engineering Department in KFUPM to give us the chance to conduct the experiments in DDS lab.

References

[1] J. Rawat, Priyanka and Singh, KamalDeep and Chaouchi, Hakima and Bonnin, "Wireless sensor networks: a survey on recent developments and potential synergies," *Journal of Supercomputing*, vol. 68, no. 1, pp. 1–48, 2014. http://dx.doi.org/10.1007/s11227-013-1021-9

[2] Basem Almadani, B. Saeed, and A. Alroubaiy, "Healthcare systems integration using Real Time Publish Subscribe (RTPS) middleware," *Computer and Electrical. Engineering*, vol. 50, pp. 67–78, Feb. 2016. http://dx.doi.org/10.1016/j.compeleceng.2015.12.009

[3] Basem Almadani, A. Shadi, and S.-H. Yang, "QoS Adaptation for Publish/Subscribe Middleware in Real-Time Dynamic Environments," *Int. Arab Journal Information Technology*, 2015.

[4] S. Oh, "Real-time performance analysis for publish/subscribe systems," *Futur. Gener. Computer System*, vol. 27, no. 9, pp. 827–323, 2010. http://dx.doi.org/10.1016/j.future.2009.09.001

[5] P. T. Eugster, P. A. Felber, R. Guerraoui, and A.-M. Kermarrec, "The many faces of publish/subscribe," *ACM Computer Survey*, vol. 35, no. 2, pp. 114–131, 2003. http://dx.doi.org/10.1145/857076.857078

[6] E. Souto, G. Guimarães, G. Vasconcelos, M. Vieira, N. Rosa, C. Ferraz, and J. Kelner, "Mires: a publish/subscribe middleware for sensor networks," *Pervasives Ubiquitous Compuing*, vol. 10, no. 1, pp. 37–44, 2006. http://dx.doi.org/10.1007/s00779-005-0038-3

[7] G. Cugola, "Using Publish / Subscribe Middleware for Mobile Systems Publish / Subscribe Middleware: A," *Communications*, vol. 6, no. 4, pp. 25–33.

[8] Basem Al-madani1 and M. F. A. Anas Al-Roubaiey, "Performance Enhancement of Limited-Bandwidth Real Time Industrial Control Systems," *J. Advanced Material. Reservoir.*, vol. 739, pp. 608–615, 2013.

[9] H. Shen, *Content-Based Publish/Subscribe Systems*. 2010.

[10] P. Boonma and J. Suzuki, "TinyDDS: An Interoperable and Configurable Publish/Subscribe Middleware for Wireless Sensor Networks," *Wireless. Technology. Concepts, Methodol. Tools Applications.*, pp. 819–846, 2011.

[11] Basem Almadani, S. Khan, M. N. Bajwa, T. R. Sheltami, and E. Shakshuki, "AVL AND MONITORING FOR MASSIVE TRAFFIC CONTROL SYSTEM OVER DDS," *Mobile Information. System Journal*, 2015.

[12] Crossbow, "TELOSB Crossbow." [Online]. Available: www.datasheetarchive.com/IRIS crossbowdatasheet.html.

[13] Crossbow, "mica2 crossbow." [Online]. Available: www.datasheetarchive.com/mica2 crossbowdatasheet.html.

[14] Crossbow, "Crossbow IRIS." [Online]. Available: www.datasheetarchive.com/IRIS crossbowdatasheet.html.

[15] Crossbow, "Crossbow micaz." [Online]. Available: www.datasheetarchive.com/micaz crossbowdatasheet.html.

[16] Anas Al-Roubaiey, "ENERGY-AWARE PUBLISH/SUBSCRIBE DDS-BASED MIDDLEWARE FOR WIRELESS SENSOR AND ACTUATOR NETWORKS," Ph.D Thesis, 2015.

[17] T. INSTRUMENTS, "CHIPCON CC2420." [Online]. Available: www.ti.com/product/cc2420.