Towards Service Level Guarantee within IoT Sensing Layer

Ahmad Khalil, Nader Mbarek, Olivier Togni
LIB, University of Bourgogne Franche-Comté
Dijon – France
emails: Ahmad.Khalil@u-bourgogne.fr, Nader.Mbarek@u-bourgogne.fr, Olivier.Togni@u-bourgogne.fr

Abstract — **Enabling service level guarantee within IoT (Internet of Things) environments is an important and a challenging task in order to enhance user experience while using IoT applications. The corresponding user service level expectations could be specified in a Service Level Agreement (SLA) that we have to conclude with the IoT Service Provider for each IoT service. As a consequence, several QoS (Quality of Service) mechanisms must be deployed within the IoT architecture layers (Sensing, Network, Cloud) to guarantee the agreed on IoT service level. We present in this paper a new QoS mechanism concerning the IoT Sensing layer. It is an adaptation of the slotted Carrier-Sense Multiple Access with Collision Avoidance (CSMA/CA) method used in the Media Access Control (MAC) layer of the IEEE 802.15.4 standard. This adaptation provides IoT smart objects with a differentiated wireless access according to the QoS class of their generated traffic in order to respect the requirements of the corresponding IoT SLA. The proposed method ensures a service level guarantee for a Low Rate Wireless Personal Area Network (LR-WPAN) in an IoT environment. Our adaptation offers a minimal delay for real time traffic along with higher Packet Delivery Ratio (PDR) for all traffics comparing to the standard slotted CSMA/CA. It consists in creating different Contention Access Periods (CAP); each will be specific for a traffic type and so for a specific QoS class. To do so, we propose firstly a QoS based wireless access method to be used by the coordinator, known as the gateway. Secondly, we propose an algorithm used by the IoT smart objects. This method, called QBAIoT (QoS Based Access for IoT environments), enables the coordinator to configure different contention periods with a specific number of slots. Consequently, the IoT objects of the same QoS class will access the channel only during their respective contention periods without collision with nodes belonging to other classes.**

Keywords - IoT; Service Level; QoS; QBAIoT; Slotted CSMA/CA; IoT Gateway; IoT objects.

I. INTRODUCTION

The Internet of Things (IoT) is currently an evidence in our daily lives. This paper extends the work conducted in [1] to show the importance of QoS guarantee in the IoT environment. In fact, by 2020, more than 20 billion digital and electronic devices will be connected resulting in an average of 2 devices per human being on Earth [2]. Thus, the impact of the IoT on human life will be important and should improve the quality of life by changing how people interact with connected objects and use IoT applications. The future growth of IoT environments will lead to an advanced technology usage enabling to facilitate the daily tasks of humans. Therefore, the improvement of the corresponding services is a major challenge within the IoT. In order to expand the usage of the IoT environment, a better user experience is expected. Consequently, QoS mechanisms should be implemented within the IoT environment [3] and especially the communication technologies used in the sensing layer of the IoT architecture such as the IEEE 802.15.4 standard [4]. The latter specifies the physical (PHY) and the Media Access Control (MAC) layers and provides an important foundation for other standards. Indeed, IEEE 802.15.4 standard is used by 6LowPAN [5] and ZigBee [6] for their lower layers implementation.

In this context, we specify QBAIoT as a novel QoS based wireless access method for IoT environments. It is an enhancement of the slotted Carrier-Sense Multiple Access with Collision Avoidance (CSMA/CA) technique, used by the IEEE 802.15.4 standard. The objective of QBAIoT is to ensure a differentiation between traffics while using the wireless channel of the IoT sensing layer. Thus, QBAIoT allows serving different IoT generated traffics while respecting the requirements of each traffic type (i.e., reduced delay for Real Time traffic). In this paper, we aim to present the design details of our proposed QoS based access method, as well as the corresponding simulation results. The reminder of the paper is organized as follows. We present in Section II the state of the art concerning the IoT environment, as well as the related technologies and we introduce the important characteristics of the IEEE 802.15.4 standard. Section III presents QoS motivations in the IoT, some related research works along with a description of an IoT Service Level Agreement (iSLA) achieved between an IoT Service Provider (IoT-SP) and an IoT Client (IoT-C). Then, we specify in Section IV our proposed method enabling QoS based access for IoT environments. Section V presents a detailed performance evaluation of our access method as well as a comparison with the standard access method. Finally, we conclude the paper in Section VI and present future works.

II. STATE OF THE ART

A. *IoT environment*

The important impact of the IoT on our society has led the international organizations to present several definitions and architectures and to create specific working and study groups focusing on IoT environments. The International Telecommunication Union - Telecommunication sector (ITU-T) presented different recommendations for the IoT such as Y.2060 [7] and Y.2066 [3] documents. Furthermore, the International Organization for standardization / International Electrotechnical Commission (ISO/IEC) presented a preliminary report about IoT in 2014 [8]. Moreover, the Internet Engineering Task Force (IETF) took an interest in the IoT environment by presenting different drafts concerning the emerging challenges for the IoT [9] [10]. Based on definitions and concepts presented by the different standardization organizations and international research projects, we can propose the following IoT definition: IoT is a system of systems interconnected via standard and interoperable communication technologies. This interconnection allows creating a considerable network of communicating objects, each addressed uniquely, in order to offer new services for improving the quality of human life. Also, self-management capabilities are essential within IoT environment in order to offer autonomous self-managed objects. In the context of the IoT, we use external resources such as cloud computing and fog computing for the processing and the storage of huge amount of data. Indeed, cloud computing functionalities enhance reliability and efficiency of IoT service provision [11]. On the other hand, fog computing decentralizes the computing capacities and distributes the operations on network extremities [12].

Different application domains with a variety of services are provided in the IoT environment. These application domains cover a wide variety of everyday services like health services, industry services, transportation services, city management services, etc. They had drawn the attention of several international organizations in order to work on standards used in the mentioned domains. For example, the ISO/IEC focuses on the standardization of underlying technologies useful in different IoT application areas. Thus, the Working Group 9 of ISO/IEC Technical Committee 1 (JTC 1/WG9) focuses on the standardization of Big Data technologies in the areas of IoT [13]. In addition, each IoT application domain attracts specific international organizations. For the e-health services, the World Health Organization (WHO) and the Program for Appropriate Technology in Health (PATH) have signed a partnership to accelerate the evolution of digital health worldwide [14]. As for the smart city domain, ISO/IEC through the technical subcommittee JTC1/SC25, standardizes microprocessor systems and interconnection mediums associated with equipment for commercial and residential environments. IoT services has attracted also, the attention of a large number of manufacturers and industrial companies like Ericsson and its partners that had offered portable prototypes for the e-health domain with long battery life [15]. In addition, Nokia offered several services and technologies on the market to manage video surveillance, sensors' networks, smart parking, etc [16].

In order to offer the IoT services, various communication technologies interconnect IoT objects and gateways within IoT environments. Each technology is suitable for a specific scenario based on different criteria such as energy consumption, CPU utilization, range of the technology, etc. IoT communication technologies correspond to an adaptation of an existing technology or to a new specifically specified technology. IoT can use wireless cellular technologies [17] (LTE, 4G, NB-IoT, 5G, etc.) or wireless non-cellular technologies (IEEE802.15.4 [4], LoRaWAN [18], ZigBee [6], 6LoWPAN [5], etc.). We describe in the following section the IEEE 802.15.4 wireless non-cellular technology, which is the foundation of our proposed QoS based access method.

B. IEEE 802.15.4

The IEEE 802.15.4 standard is an IEEE proposed standard for Low Rate Wireless Personal Area Networks (LR-WPAN). It defines the physical and the MAC layers to provide a basic format. This format will be used by other technologies and protocols by adding their own specificities through the specification of the higher layers. The IEEE 802.15.4 physical layer specifies different essential parameters: 250 Kbit/s of data rate for a 2.4 GHz band, control functions like the activation or deactivation of the radio module, the test of the channel occupation and the choice of the transmission channel. On the other hand, the MAC layer defines the data management format and specifies the usage of different access methods for the wireless shared channel (i.e., Unslotted CSMA/CA, Slotted CSMA/CA, TSCH CCA, TSCH CSMA/CA, CSMA/CA with PCA, DSME, etc.). As for data encryption, the IEEE 802.15.4 standard uses AES-128 (Advanced Encryption Standard) to ensure data confidentiality [4]. Different standards use IEEE 802.15.4 as a foundation for their lower layers. We can mention as an example the 6LowPAN standard that combines IPv6 with low power WPAN networks. Another example is ZigBee, a specification for a series of high-level, low-power communication.

IEEE 802.15.4 supports a beacon-enabled mode using a superframe structure, which is the base of our contribution. The superframe (see Fig. 1) consists of an active part known as the Superframe Duration (*SD*) and can be followed by an inactive period. The active part is formed by 16 equally sized time slots partitioned into a Contention Access Period (*CAP*) where nodes compete to gain the access to the channel; and an optional Contention Free Period (*CFP*) where nodes are allocated guaranteed time slots.

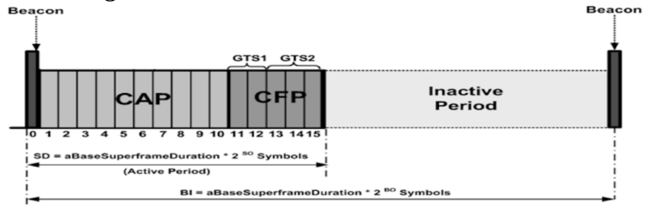

Figure 1. IEEE 802.15.4 beacon enabled mode superframe structure

In beacon-enabled mode, the coordinator sends periodically a beacon frame on the network including all the superframe specifications. The beacon, sent at the Beacon Interval (*BI*) time, allows the coordinator to identify its WPAN and ensures that all the objects are synchronized. The Beacon Order (*BO*) and Superframe Order (SO) parameters determine the Beacon Interval (*BI*) and *SD*, respectively as mentioned in (1) and (2). The Base Superframe Duration (*BSFD*) corresponds to the minimum duration of the superframe (*SO* = 0).

$$BI = BSFD * 2^{BO} \quad (1)$$

$$SD = BSFD * 2^{SO} \quad (2)$$

BSFD is fixed to 960 symbols of 4 bits or 15.36 ms assuming the data rate of 250 Kbit/s for the 2.4 GHz band. In addition, *BO* and SO should respect the inequality $0 \leq SO \leq BO \leq 14$ [4].

Three variable are used in the slotted CSMA/CA algorithm (see Fig. 2): the Backoff Exponent (*BE*), the Contention Window (*CW*) and the Number of Backoffs (*NB*). To compute the backoff delay, that an object has to observe before performing the Clear Channel Assessment (*CCA*), the algorithm chooses a random value for the backoff delay between 0 and ($2^{BE} - 1$). CW is the number of backoff periods during which the channel must be idle before accessing the channel. By default, the value of *CW* is fixed to 2. *NB* is the number of backoff executed for channel access. This value is initialized to 0 and is compared to a maximum value, *macMaxCSMABackoffs* by default equal to 5. In case the *NB* value is greater than this maximum value, a failure occurs.

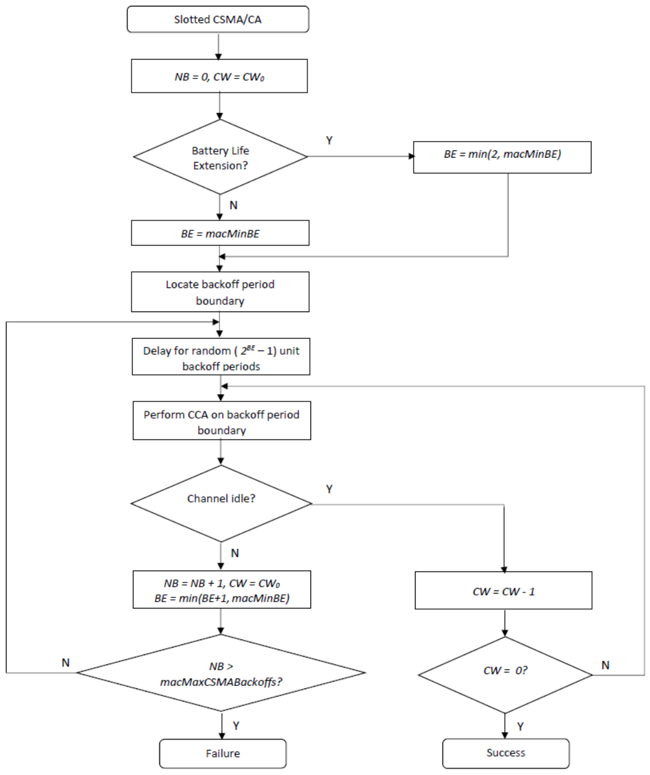

Figure 2. Slotted CSMA/CA Algorithm

The slotted CSMA/CA algorithm is activated for each transmission of a new packet and is executed during the *CAP* as follows [4]:

- *NB* and *CW* are initialized
- If the battery life extension is true, *BE* is initialized to the minimum between 2 and *macMinBE* (by default 3). If the battery life extension parameter is fixed to false, *BE* is initialized to 2
- The node using the algorithm waits the backoff delay, and then performs *CCA*
- If the channel is busy, *CW* is re-initialized to 2, *NB* and *BE* are incremented. *BE* must not exceed *aMaxBE* (by default 5). If *macMaxCSMABackoffs* is reached, the algorithm reports a failure to the higher layer. If *NB* < *macMaxCSMABackoffs*, the backoff operation is restarted and the *CCA* should be performed again
- If the channel is sensed idle and *CW* > 0, the *CCA* is repeated and *CW* decremented. Otherwise, the node attempts to transmit if the remaining time in the current *CAP* is sufficient to transmit the frame and receive the acknowledgement. If not, the process is deferred to the next superframe.

III. QoS GUARANTEE IN THE IoT

A. *Motivations and challenges for QoS guarantee in the IoT*

The ITU-T E.800 [19] has defined QoS as the totality of the characteristics of a telecommunication service to satisfy in order to meet the user requirements. In this context, a QoS requirement is expressed in terms of QoS parameters (Delay, Jitter, Packet Delivery Ratio, Effective Data Rate, etc.). QoS guarantee in the IoT environment requires an effective and optimized management of the corresponding resources to improve users' experience. In order to provide predictable services, QoS mechanisms in the IoT environment handle delays, jitter, bandwidth and packet loss ratio by classifying traffic. As the IoT environment is made of different technologies and heterogeneous networks, different types of data and streams exist on a single system. Hence, it is important to provide the IoT environment with QoS guarantee mechanisms to meet the requirements of each type of traffic [9]. QoS guarantee is a critical challenge in the IoT, as the number of connected objects increases considerably leading to a greater amount of created and transported data with different characteristics. Consequently, the performance of the IoT system will be affected and especially QoS constrained data traffic due to congestion periods. Deploying QoS mechanisms within IoT environment will enhance the performance by identifying and differentiating traffic in order to allow a reduced cost and a better scalability [10].

The importance of the QoS guarantee in the IoT has been put forward by various international organizations The ITU-T describes the importance of QoS integration in the IoT through various documents such as Y.2066 [3] where it was mentioned that service priority is an important requirement. In addition, Y.2066 indicates that the prioritization functionality satisfies different service requirements of IoT users. On the other hand, LinkLabs, an American company developing technologies for computer networks, indicates that integrating QoS into IoT allows a better management of the corresponding capabilities and resources in order to provide a reliable and optimized infrastructure for connecting objects. According to LinkLabs, QoS mechanisms enables predictable IoT services thanks to better delay, jitter, bandwidth and Packet Delivery Ratio (PDR) by classifying traffic and offering services according to systems' resources [22].

In order to provide QoS within an IoT architecture, the requirements of each layer (Sensing layer, Network layer and Cloud Layer) should be addressed through one or several mechanisms. The Sensing layer includes all the IoT objects along with the gateways allowing their interconnection and management. Thus, the QoS provision at this layer should meet the IoT objects and gateways requirements. An essential challenge for this layer is traffic differentiation and prioritization. It can be offered by classifying the different flows according to their criticality and applying prioritization through different adapted QoS mechanisms. Thus, it is important to classify IoT applications according to specific criteria in order to propose an appropriate QoS mechanism while respecting their traffics characteristics. Each set of applications will have mechanisms well adapted to their requirements. In addition, at this layer the optimization of the systems resources usage should be applied in order to offer the best performances. The network layer of the IoT architecture includes all network features such as routing, handoff, and path management (path selection and recovery) through a multi-path infrastructure. This layer acts as a network infrastructure interconnecting the Sensing layer to the Cloud layer. The integration of QoS mechanisms in this layer should consider the large number of requests and data transiting from the Sensing layer to the Cloud layer of the IoT architecture. The data processing must be differentiated in the Network layer. It must prioritize requests according to their importance. As a result, the QoS requirements of the IoT Network layer correspond to the traditional QoS requirements of a network infrastructure while adapting these needs to the characteristics of the IoT environment. Finally, the IoT Cloud layer includes computing and storage capabilities. In addition, this layer hosts IoT applications enabling processing data for useful purposes. The QoS guarantee in the Cloud layer is an emerging discipline with several research challenges. This is due to the lack of standardized end-to-end approaches for QoS assurance and the existence of various constraints and QoS parameters specific to each cloud service. Indeed, QoS requirements in the Cloud layer depend on the provided service (Infrastructure as a Service - IaaS, Software as a Service - SaaS, Platform as a Service - PaaS) by the Cloud Service Provider (CSP). Finally, it is necessary to specify the needs and mechanisms ensuring end-to-end QoS guarantee across the different layers of the IoT architecture. This end-to-end QoS provision allows customers to perceive the requested service level without distinguishing the declination of this QoS according to the IoT architecture several layers.

In the next sections, we present related research work concerning QoS offer in IoT environments and we describe the IoT Service Level Agreement.

B. Related research work

Different international projects and research works had studied the Quality of Service in the IoT environment and its impact on the service provision. The European project OpenIoT [23] specified different QoS parameters and metrics for the IoT. These metrics include utility metrics related to sensors and other metrics related to the network and application. As an example of utility metrics, OpenIoT indicated the Quality of sensors that determines the accuracy of measurement, the energy consumption, data volume, and bandwidth. For the other metrics, system lifetime is taken into consideration. In addition, traditional QoS parameters are used such as latency, jitter, delay, throughput, etc. On the other hand, this project presented a high level architecture based on a QoS Manager that keeps track of the following parameters: quality of sensors, energy consumption, trustworthiness, bandwidth and data volume.

The research work carried out in [24], concerning the guarantee of QoS in IoT, proposes to classify various IoT applications according to 3 service models (i.e., Open Service Model, Supple Service Model, Complete Service Model).. It maps each class to a physical topology for sensors' implementation. Open Service Model corresponds to interactive, non-real-time and non-critical applications. Supple Service Model corresponds to interactive, Soft Real Time and critical applications. Complete Service Model corresponds to interactive, Hard Real time and critical applications. Thus, the authors classified the IoT applications belonging to different domains according to these 3 models. In addition, this work has matched the proposed service models with physical topologies (star topology and random topology) at the device layer to meet the needs of each model. Indeed, the applications belonging to the Complete model must be provided through a physical star topology to obtain better delays. On the other hand, applications belonging to the Open model must be provided through a random physical topology for better energy consumption.

Furthermore, other research works had focused on the QoS in the lower layer of the IoT architecture (sensor layer). For example, the research work conducted in [25], tried to use different queues and a scheduler to ensure a certain priority for QoS constrained flows. Moreover, different research work tried to adapt the slotted CSMA/CA algorithm to ensure QoS guarantee. Thus, the authors present in [26] a contribution that allows the delivery of critical data with a highest priority during the CFP. In [27], the authors describe the usage of different values for *CW*, *minBE* and *maxBE* to differentiate services thanks to three different priority levels. However, these research works did not take into consideration the existence of real time applications in the IoT environment requiring a reduced delay that does not exceed milliseconds range. For this matter, our proposed QoS based access method aims to provide a differentiation between IoT objects' flows based on different QoS classes' characteristics.

C. IoT Service Level Agreement

In this research work we consider four types of traffics corresponding to four QoS classes as specified in a previous work [28]: Real Time Mission Critical (RTMC), Real Time Non Mission Critical (RTNMC), Streaming and Non Real Time (NRT). Each QoS class corresponds to several requirements regarding performance parameters such as delay, jitter, etc. For example, our specified Real Time QoS classes are more sensitive to delay and jitter variation. The Streaming class is more sensitive to jitter variation while the Non Real Time class is a non-constrained QoS traffic class.

In order to specify the concrete requirements of each QoS class (IoT-C's expected value of each performance parameter), we had presented in our previous work [28] a specific Service Level Agreement (SLA) for IoT environments, called iSLA, in order to allow an IoT-SP and an IoT-C to negotiate and agree on the expected service level. The expectations are described through different measurable parameters according to the IoT type of service (i.e., QoS class). We specify for each QoS class a set of measurable parameters that are critical for the type of data concerned by that QoS class. In addition, the IoT-SP uses a cloud infrastructure, a network infrastructure and a sensing infrastructure to provide the IoT service. In this context, our proposed iSLA considers the characteristics of each sub-infrastructure needed by the provided IoT service. Thus, the corresponding sub-SLAs, forming the global iSLA, are concluded with a CSP (i.e., cloud SLA: cSLA) and a Network Service Provider (NSP) (i.e., network SLA: nSLA). For the sensing infrastructure, the IoT- SP dispose of two kinds of gateways; High Level Gateways (HL-Gws) used for self-management provision and Low Level Gateways (LL-Gws) used to collect data from IoT objects. The IoT-SP concludes another internal sub-SLA called the gateway SLA (gSLA) to specify the characteristics of the gateways for the corresponding IoT Service. The gSLAs (stored on the HL-Gw) allow the HL-Gw to have detailed information concerning the characteristics of the underlying infrastructure for self-management consideration. After concluding the cSLA, nSLA and gSLA, the IoT-SP is able to conclude the global iSLA with the IoT-C. In order to describe the iSLA establishment process accomplished by the IoT-SP, we specify a Finite State Machine (FSM) diagram with several states illustrating the behavior of the IoT service Provider (see Fig. 3).

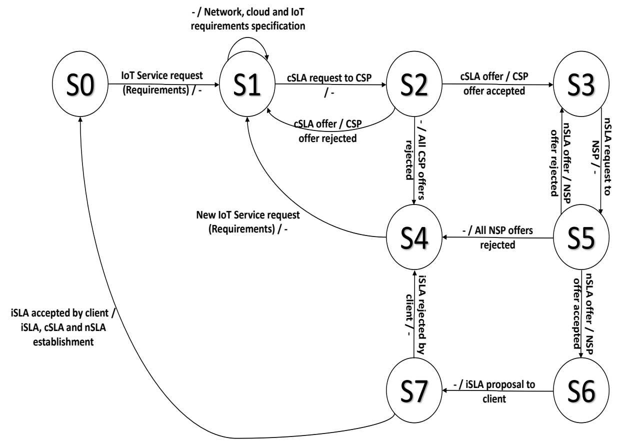
- / Network, cloud and IoT requirements specification
S0
IoT Service request (Requirements) / -
S1
cSLA request to CSP / -
S2
cSLA offer / CSP offer accepted
S3
cSLA offer / CSP offer rejected
- / All CSP offers rejected
nSLA offer / NSP offer rejected
nSLA request to NSP / -
New IoT Service request (Requirements) / -
S4
- / All NSP offers rejected
S5
iSLA accepted by client / iSLA, cSLA and nSLA establishment
iSLA rejected by client / -
nSLA offer / NSP offer accepted
S7
- / iSLA proposal to client
S6

Figure 3. Finite State Machine of iSLA establishment

In state S0, the IoT-SP waits for the service requirements from the client to start the process of iSLA establishment. After receiving these requirements, the IoT-SP classifies the requirements in state S1 and changes to state S2 when it sends the cSLA request to the CSP in order to conclude a cloud SLA. The CSP sends to the IoT-SP a cSLA offer. If the offer is rejected, then the IoT-SP state changes again to state S1 but if the offer is accepted, the IoT-SP reaches state S3. If all CSP offers are rejected, the IoT-SP will be at state S1 after reaching state S4 to wait for a new set of requirements as the older set cannot be satisfied and the process restarts. The same process is executed with the NSP. If the NSP offer is accepted, the IoT-SP will be at the state S6. If all NSP offers are rejected, the IoT-SP will be at state S1 after reaching state S4 to wait for a new set of requirements as the older set cannot be satisfied and the process restarts. After accepting the nSLA, the IoT-SP at state S6 sends an iSLA proposal to the client and reaches state S7. If the client rejects the iSLA, the IoT-SP passes to state S4 and a new round of negotiation with the service provider should be achieved in order to build a new iSLA. If the iSLA proposal is accepted, the IoT-SP concludes the sub-SLAs with the corresponding NSP and CSP and concludes the iSLA with the IoT-C while reaching the initial state S0.

We specify in the next section our proposed QoS mechanism called QBAIoT. It is a wireless access method based on the the four QoS classes mentioned above and ensures a differentiation in traffic processing for QoS integration within the sensing layer of the IoT architecture.

IV. QoS Based Access for IoT

We describe in the following our QoS based access method for IoT environments called QBAIoT. The specification of our novel access method is based on a new superframe structure, as well as algorithms implemented within the IoT Gateway and IoT objects enabling Class based Contention Free Periods.

A. Class based Contention Free Period Access

Our proposed access method consists in using an IEEE 802.15.4 superframe that respects the requirements of the four QoS classes. For achieving our QoS guarantee according to the requirements of the different traffics, we adapt the structure of the IEEE 802.15.4 superframe in order to include a CAP (called QoS CAP) for each traffic corresponding to a specific QoS class. Moreover, there are no CFP and inactive periods in our adapted superframe.

We had removed the inactive period to reduce the delay of Real Time generated data. In this context, we can find up to four QoS CAPs in our superframe in case the IoT gateway (Coordinator or LL-Gw) is configured with four QoS classes (see Fig. 4).

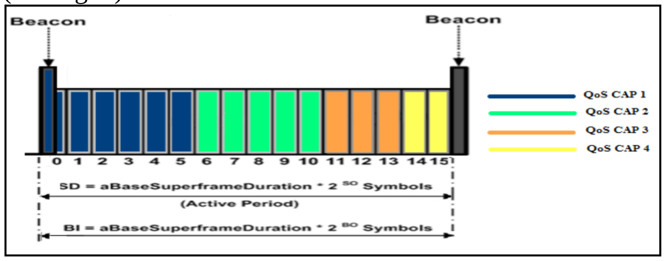

Figure 4. QBAIoT superframe structure

During each QoS CAP, only objects belonging to the corresponding QoS class can try to use the slots in order to send their data. The slots configuration and the number of QoS CAPs in the superframe is based on the number of QoS classes available in the IoT gateway environment. Different configurations for the superframe based on the existence of

Real Time applications and the number of QoS classes in the considered IoT environment are possible. If the network includes one QoS class, a single CAP will exist in the superframe and the normal IEEE 802.15.4 slotted CSMA/CA algorithm is used. If there are multiple QoS classes with a minimum of one Real Time class in the network, BO and SO will be configured with the value 2 in order to minimize the latency of Real Time traffic thanks to a reduced Superframe Duration among others. Consequently, based on (1) and (2), BI and SD correspond to 61.44 ms with a slot time of 3.84 ms. If multiple QoS classes exist with no Real Time classes, BO and SO are set to 3 fixing BI and SD to 122.88 ms with a slot time of 7.68 ms. We specify for each QoS CAP a fixed number of slots. This configuration differs according to the number of existing QoS classes in the IoT Gateway environment. For example, in the case of 4 QoS classes the superframe slot configuration is as follows: RTMC class QoS CAP is allocated 6 slots, RTNMC class QoS CAP is allocated 5 slots, Streaming class QoS CAP is allocated 3 slots and NRT class QoS CAP is allocated 2 slots. So, slots configuration and the number of QoS CAP in the superframe is based on the number of existing QoS classes.

B. IoT Gateway QoS based access method design

For the coordinator part (i.e., IoT Gateway) of our proposed QBAIoT access method, we specify Algorithm 1 (see Fig. 5) among with the corresponding variables described in Table I.

Algorithm 1 Gateway QBAIoT Access Method Algorithm

```
Input:  Nb_QoS_Classes, RT_Classes
1:  N ← 1
2:  if (Nb_QoS_Classes = 1) then
3:       BO, SO ← 14
4:       MAC ← Slotted_CSMA
5:       While true do
6:            Send_Beacon (BO, SO, CAP)
7:            Receive_Data ()
8:       end while
9:  else
10:      if (RT_Classes = 0) then
11:           BO, SO ← 3
12:           MAC ← QBAIoT
13:           Initial_Slots_Configuration ()
14:           While true do
15:                Send_Beacon (BO, SO, QoS CAPs)
16:                While(N<=Nb_QoS_Classes) do
17:                     Receive_Data (QoS CAP)
18:                     N ← N + 1 // Next QoS CAP
19:                end while
20:           end while
21:      else
22:           BO, SO ← 2
23:           MAC ← QBAIoT
24:           Initial_Slots_Configuration ()
25:           While true do
26:                Send_Beacon (BO, SO, QoS CAPs)
27:                While(N<=Nb_QoS_Classes) do
28:                     Receive_Data (QoS CAP)
29:                     N ← N + 1 // Next QoS CAP
30:                end while
31:           end while
32:      end if
33: end if
```

Figure 5. Gateway QBAIoT Access Metthod Algorithm

TABLE I. VARIABLE SPECIFICATION OF ALGORITHM 1

Name of the variable	Description
Nb_QoS_Classes	Number of QoS classes
RT_Classes	Number of Real Time classes
N	Index of QoS classes
MAC	Channel access algorithm
QoS CAP; CAP	Configuration of the CAP (CAPStart and CAPEnd)
Initial_Slots_Configuration()	Algorithm that computes the slots configuration based on the Number of QoS classes and Number of Real Time classes.

As shown in Fig. 6, **t**he IoT Gateway using our QoS based access method (i.e., QBAIoT gateway) will receive data from objects during the corresponding QoS CAPs.

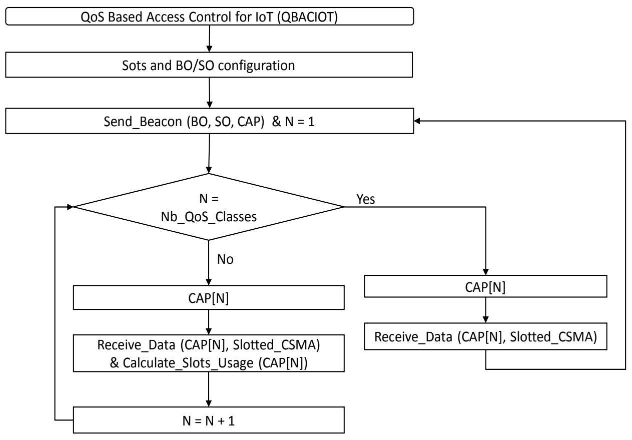
QoS Based Access Control for IoT (QBACIOT)
Sots and BO/SO configuration
Send_Beacon (BO, SO, CAP) & N = 1
N = Nb_QoS_Classes
Yes
No
CAP[N]
Receive_Data (CAP[N], Slotted_CSMA) & Calculate_Slots_Usage (CAP[N])
N = N + 1
CAP[N]
Receive_Data (CAP[N], Slotted_CSMA)

Figure 6. Gateway QBAIoT Access method

At each Beacon Interval, the gateway sends the beacon including the information regarding the values of *BO*, *SO* and the first and final slot for each QoS CAP. These values are used by the IoT objects to calculate the slot time and to determine during which time they are allowed to compete for the channel. A QBAIoT gateway should include also self-management capabilities.

A self-configuring capability enables the gateway to adapt the superframe slots configuration according to the existing number of QoS classes within its environment. A self-optimizing capability is performed in case of unused slots in a QoS CAP thanks to a slot reallocation mechanism covering the entire superframe. The self-management capabilities design is out of the scope of this paper.

C. *Class based access for IoT objects*

For the IoT object part of our proposed QBAIoT access method, we specify Algorithm 2 (see Fig. 7) among with the corresponding variables described in Table II.

Algorithm 2 Object QBAIoT Access Method Algorithm

```
1: Receive_Beacon (BO, SO, QoS CAPs)
2: Configuration (BO, SO, QoS CAPs)
3: while (Slot ∈  [CAPStart, CAPEnd] and Data = true) do
4:         if  (Slotted_CSMA (Slot) = Success) then
5:                 Send_Data (Success, PAN Coordinator)
// slotted CSMA/CA returns a success state
6:         else
7:                 Send_Data (Failure, PAN Coordinator)
// slotted CSMA/CA returns a failure state
8:         end if
9: end while
10: if (Slot < CAPStart) then
11:         Wait_until (Slot ∈ [CAPStart, CAPEnd])
12: else
13:         Wait_Until (Beacon) // Wait until next superframe
14: end if
```

Figure 7. Object QBAIoT Access Method Algorithm

TABLE II. VARIABLE SPECIFICATION OF ALGORITHM 2

Name of the variable	Description
QoS CAP	Configuration of the CAP (CAPStart and CAPEnd)
CAP_Start_Slot	The first slot for the corresponding QoS CAP assigned to the object
CAP_End_Slot	The last slot for the corresponding QoS CAP assigned to the object

Any object in the IoT Gateway environment receives the beacon. According to the QoS class it belongs to, the object will determine during which QoS CAP it can compete to access the shared medium. When an IoT object generates data, it should test if it has the right to compete in order to send its traffic. If the corresponding QoS CAP of the object has not started, it waits until its CAP time and then competes to send the data according to our adapted slotted CSMA/CA algorithm. If the object QoS CAP had passed, it should wait until the corresponding QoS CAP in the next SuperFrame.

Fig. 8 shows the adapted CSMA/CA algorithm adopted by the IoT Objects that communicate using our QBAIoT method.

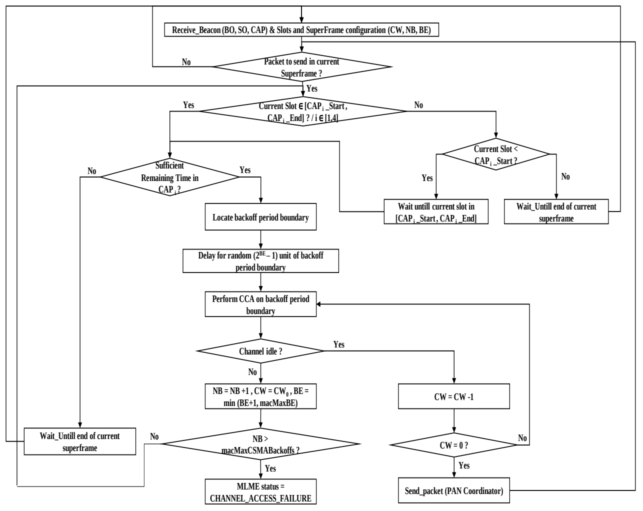
Receive_Beacon (BO, SO, CAP) & Slots and SuperFrame configuration (CW, NB, BE)
Packet to send in current Superframe ?
No
Yes
Current Slot ∈ [CAP$_i$_Start, CAP$_i$_End] ? / i ∈ [1,4]
Yes
No
Current Slot < CAP$_i$_Start ?
Sufficient Remaining Time in CAP$_i$?
No
Yes
Yes
No
Locate backoff period boundary
Wait untill current slot in [CAP$_i$_Start, CAP$_i$_End]
Wait_Untill end of current superframe
Delay for random (2^{BE} – 1) unit of backoff period boundary
Perform CCA on backoff period boundary
Channel idle ?
Yes
No
NB = NB +1 , CW = CW$_0$, BE = min (BE+1, macMaxBE)
CW = CW -1
Wait_Untill end of current superframe
No
NB > macMaxCSMABackoffs ?
CW = 0 ?
No
Yes
Yes
MLME status = CHANNEL_ACCESS_FAILURE
Send_packet (PAN Coordinator)

Figure 8. Object QBAIoT Access Method

V. PERFORMANCE EVALUATION AND RESULTS

A. *Simulation environment*

In order to evaluate our proposed QBAIoT access method, we conduct a simulation study using OMNeT++ based on the IEEE 802.15.4 model [29] including all the necessary features like the beacon, the superframe structure, etc. We had adapted this model to take into consideration our proposed QoS based access method thanks to a superframe with no CFP and different QoS CAPs. In our simulation scenario, we simulated four QoS classes (RTMC, RTNMC, Streaming and NRT). We used a star topology with a single coordinator (i.e., IoT Gateway) where all devices (i.e., IoT objects) are in each other's radio range. Each device transmits data to the coordinator. The data packets are generated periodically but are transmitted during the corresponding QoS CAP. Table III shows the used simulation parameters.

In the first simulation scenario, we fixed the Data Generation Interval (DGI) to 0.25 seconds and we increased the number of IoT objects from 4 (1 per QoS class) to 12 (3 per QoS class). The IoT objects are sending data simultaneously as they start generating data at the same time with the same interval of packet generation. As for the second set of simulations, we used a DGI of 0.125s allowing generating a double amount of packets comparing to the first set of simulations.

TABLE III. SIMULATION PARAMETERS

Parameter	Value
Carrier Frequency	2.4 GHz
Transmitter Power	1 mW
Bit rate	250 Kbps
Simulation Time	100 s
Max Frame Retries	3
Mac Payload Size	50 Bytes

B. *Performance evaluation*

The evaluation of our proposed QoS based access method is based on different performance parameters concerning the traffic of our QoS classes. The importance of these parameters depends on the characteristics of the corresponding traffic. Indeed, the average delay is very important and critical for the RTMC and RTNMC traffic whereas it is less important for Streaming traffic and not important for NRT traffic. In this context, we considered the following performance parameters.

- Average Delay: It refers to the average time experienced by a generated packet to be received by the destination. It is computed by dividing the total delay experienced for all the packets by the number of packets as shown in equation (3).

$$Average\ Delay = \frac{\Sigma\ Delay}{Number\ of\ received\ packets} \qquad (3)$$

- PDR: It expresses the degree of reliability achieved by the system for successful transmissions. It is obtained by dividing the number of received packets by the number of generated packets as shown in equation (4). Non received packets are either lost due to a collision or still in the sender buffer waiting for channel access.

$$PDR = \frac{Number\ of\ received\ packets}{Number\ of\ sent\ packets} \qquad (4)$$

- Mean Packet Delivery Ratio (MPDR): It expresses the degree of reliability achieved by the system for successful transmissions of all traffic types. It is obtained by computing the mean value of the PDRs of the different traffic types as shown in equation (5).

$$MPDR = \frac{\Sigma\ PDR}{Number\ of\ traffic\ types} \qquad (5)$$

- Effective data rate (EDR): It evaluates the link bandwidth utilization. It is computed by multiplying the number of received packets by their sizes to obtain the total length of the frame, which is divided by the simulation time as shown in equation (6).

$$EDR = \frac{\text{Number of received packets} * \text{Packet Size}}{\text{Simulation Time}} \qquad (6)$$

Table IV presents the delay evaluation for 4 QoS classes traffic while using our proposed QBAIoT access method and the traditional IEEE 802.15.4 slotted CSMA/CA method for the first set of simulations (using the 0.25s DGI). The Delay QoS parameter is very sensitive for RTMC and RTNMC traffic. The obtained results in Table IV shows that for 4 objects (1 object per QoS CAP), our proposed method enables better delay for the RTMC traffic (10 ms less than the standard) and the RTNMC traffic (7 ms less than the standard). This difference becomes greater while increasing the number of objects. For 8 objects in the IoT environment (2 objects per QoS CAP), we can observe a 35 ms better delay for RTMC traffic and 26 ms better delay for RTNMC traffic. The better delays that we obtain for Real Time traffic with our proposed method are owing to the fact of giving the Real Time classes a more important number of slots in which they can send their data without any collision with other objects belonging to other non-real time QoS classes. Consequently, data packets do not need to wait in buffer for a long time. They are served faster than other traffic types.

Although it is not critical for NRT traffic, we notice important delays for this traffic when the total number of objects is equal to 12 (3 objects per QoS CAP). This delay comes from the fact that this traffic is served during 2 slots in each superframe and that each traffic class generates the same number of packets in our scenario at the same time; all packets of the different QoS classes are generated at the same time. So, when the number of objects in the NRT class increases, the delay will increase because the generated traffic is greater than the allocated capacity of 2 slots resulting in a great number of packets in the sending buffer.

TABLE IV. AVERAGE DELAY EVALUATION FOR DIFFERENT TRAFFIC TYPES USING QBAIOT AND IEEE 802.15.4 STANDARD

	Delay in seconds							
QoS class / Number of object per class	RTMC Standard	RTMC QBAIoT	RTNMC Standard	RTNMC QBAIoT	Streaming Standard	Streaming QBAIoT	NRT Standard	NRT QBAIoT
1	0.062	0.052	0.063	0.056	0.062	0.063	0.067	0.067
2	0.1	0.065	0.1004	0.074	0.102	0.104	0.104	0.67
3	0.115	0.09	0.123	0.106	0.0129	0.124	0.131	30.61

Table V shows the Packet Delivery Ratio for 4 QoS classes traffic while using our proposed QoS based access method and the IEEE 802.15.4 standard for the first set of simulations. Our QBAIoT access method is giving, for all QoS classes three times better PDR with one object by class, four times better PDR with two objects by class and 6 times better PDR (except NRT class 1,5 times) with 3 objects by class than IEEE 802.15.4 standard method. We obtain a better PDR with our approach thanks to an optimized channel access per class avoiding collisions between different QoS classes. Indeed, for each QoS CAP, only objects of the corresponding QoS class can compete to access the channel. For example, with 1 object per QoS class, there is no competition between objects to gain access to the channel during each slot with QBAIoT comparing to a competition between 4 objects while using IEEE 802.15.4. Consequently, with QBAIoT a lower number of objects are competing for accessing the channel for a given slot. Packets will not run the slotted CSMA/CA algorithm for several times and there is no need to drop packets after several attempts when macMaxCSMABackoffs is reached.

TABLE V. PDR EVALUATION FOR DIFFERENT TRAFFIC TYPES USING QBAIoT AND IEEE 802.15.4 STANDARD

	Packet Delivery Rate							
QoS class / Number of object per class	RTMC Standard	RTMC QBAIoT	RTNMC Standard	RTNMC QBAIoT	Streaming Standard	Streaming QBAIoT	NRT Standard	NRT QBAIoT
1	0.31	1	0.36	1	0.35	1	0.33	1
2	0.18	0.99	0.21	0.99	0.21	0.97	0.19	1
3	0.20	0.98	0.18	0.96	0.15	0.90	0.16	0.26

As for the effective data rate, Table VI compares the obtained results using our proposed QBAIoT method and the traditional slotted CSMA/CA of the IEEE 802.15.4. The obtained results show that QBAIoT allows always better effective data rate than the traditional approach, as the PDR of QBAIoT is always higher. A lower number of collisions offer a higher number of received packets. Consequently, the number of bits served is higher during the simulation time allowing a greater EDR with QBAIoT. We can note an average of 4 times better EDR with QBAIoT comparing to IEEE 802.15.4 for all QoS classes with 4, 8 and 12 objects in the IoT environment (except for the NRT traffic with 3 objects per QoS class in the environment where the EDR with QBAIoT is only 1.7 time better).

TABLE VI. EDR EVALUATION FOR DIFFERENT TRAFFIC TYPES USING QBAIoT AND IEEE 802.15.4 STANDARD

	Effective Data Rate in bits per second							
QoS class / Number of object per class	RTMC Standard	RTMC QBAIoT	RTNMC Standard	RTNMC QBAIoT	Streaming Standard	Streaming QBAIoT	NRT Standard	NRT QBAIoT
1	508	1600	588	1600	564	1600	528	1600
2	604	3190	680	3180	692	3120	612	3200
3	972	4710	904	4620	728	4330	722	1240

In the second set of simulations, we used the same environment as for the first set but with a Data Generation Interval of 0.125s allowing generating 800 packets per objects during the simulation time.

Fig. 9 presents the comparison of the different average delay results concerning RTMC traffics while using QBAIoT and IEEE 802.15.4 standard with a DGI of 0.125s, as well as with 1, 2 and 3 objects per QoS class. We can note that with QBAIoT, better average delays are observed in all cases. By incrementing the number of objects, the results of average delay turn into greater values as more important number of packets should be served during the same QoS CAP. Comparing to Table IV, the observed average delay by RTMC traffic becomes more important by decreasing the DGI. Indeed, lower DGI values correspond to a more important number of generated packets each second.

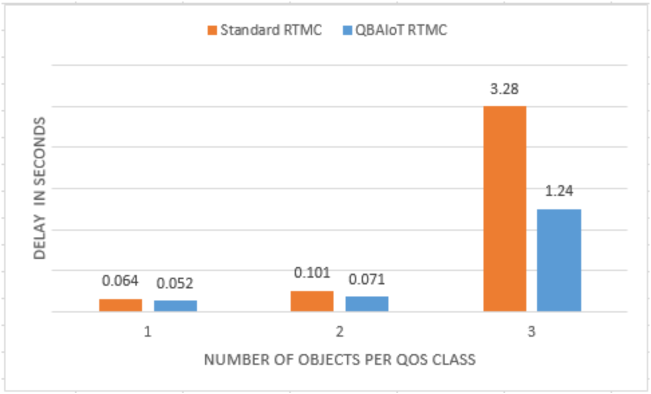

Figure 9. Average delay evaluation for RTMC traffic using QBAIoT and IEEE 802.15.4 for a DGI of 0.125s

Fig. 10 presents a comparison of QBAIoT RTMC traffic average delay for a DGI of 0.125s and 0.25s for 1, 2 and 3 objects per QoS class. We can note that the generation of the same number of packets by a single object allows observing a lower average delay comparing to the same number of packets generated by two or more objects. For instance, the generation of 800 RTMC packets by a single object (1 object per QoS class with a DGI of 0.125s) induces a 0.052 ms average delay for RTMC traffic. Whereas, the generation of 800 RTMC packets by two objects (2 objects per QoS class with a DGI of 0.25s) induces an average delay of 0.065 ms for RTMC traffic. The 13 ms higher average delay with two objects generating the 800 packets is due to the collisions that can occur between the two objects of the same QoS class during the contention for accessing the channel.

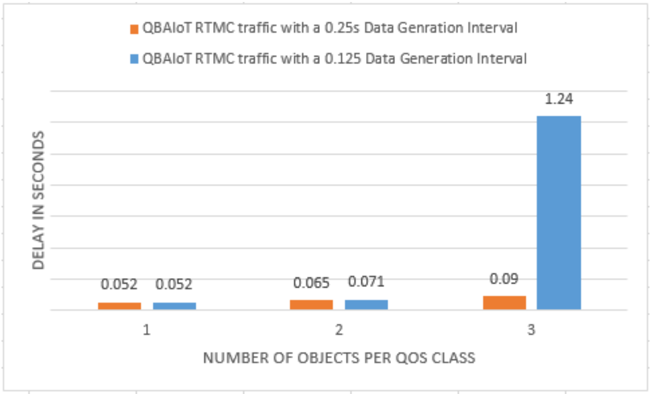

Figure 10. Average delay evaluation for QBAIoT RTMC traffic with different DGI

Fig. 11 presents the comparison of the different average delay results of RTNMC traffics while using QBAIoT and IEEE 802.15.4 standard with a DGI of 0.125s, as well as 1, 2 and 3 objects per QoS class. We can note that with QBAIoT a better dealy is observed by RTNMC traffic in all cases thanks to the fact of minimizing the collisions and organizing the time during which each object can compete to gain access to the shared medium.

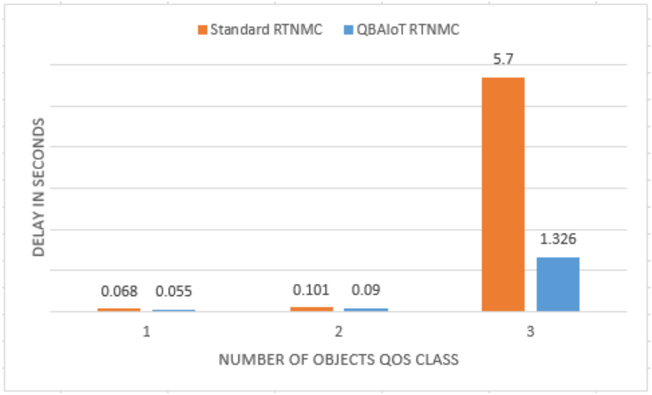

Figure 11. Average delay evaluation for RTMC traffic using QBAIoT and IEEE 802.15.4 for a DGI of 0.125s

Fig. 12 presents the comparison between the different MPDR values of the different traffics with the DGIs of 0.25s and 0.125s while using QBAIoT. We can observe that with 1 object per QoS class, as there is no collisions between packets of different objects and the maximum capacity of the medium has not been reached yet for each QoS CAP, the PDR mean value is equal to the maximum value of 1. As for the case with 2 and 3 objects per QoS class, the MPDR value is lower with a DGI of 0.125s as the number of competition executed to access the channel is higher than the case with a DGI of 0.25s. Consequently, the probability of having a collision is higher resulting in lower MPDR values.

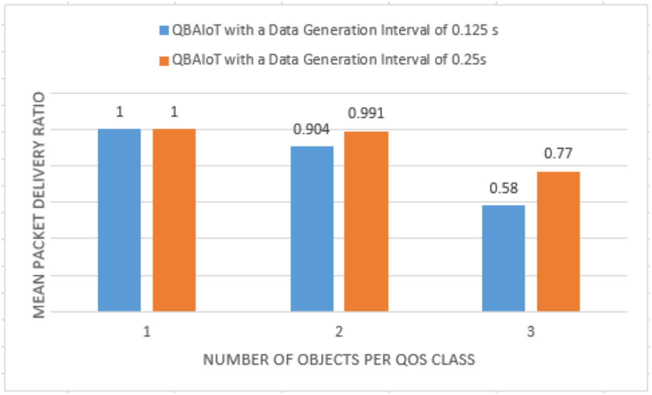

Figure 12. Mean Packet Delivery ration for different DGIs

VI. CONCLUSION

To ensure better user experience in the IoT environment, researchers try to optimize the delivered services while guaranteeing the QoS. Different access technologies could be used in the sensing layer of the IoT architecture. Several of these technologies are based on the IEEE 802.15.4 standard but the latter does not provide any QoS guarantee for the traffic generated by objects using this standard to access the IoT infrastructure. Therefore, we proposed the QBAIoT access method as an enhancement of the IEEE 802.15.4 slotted CSMA/CA mechanism in order to take into consideration QoS requirements of 4 different kinds of QoS traffic classes generated in the IoT environment. QBAIoT allows to respect the service level negotiated between the IoT-C and the IoT-SP during the establishment of the iSLA. In particular, QBAIoT QoS provision within the lower layer of the IoT architecture (Sensing Layer). We compared our proposed access method to the IEEE 802.15.4 standard and we showed that we obtain better results while using our QoS based access method to guarantee a reduced delay for Real Time traffic, as well as a greater PDR and effective data rate for all QoS classes with different DGIs.

As ongoing work, we aim to provide the IoT environment with a self-configuring capability allowing activating the minimum needed number of objects per QoS class in an autonomic manner while optimizing energy consumption. To do so, we will use the Fuzzy Logic theory in order to let the system choose autonomously the best objects in order to minimize the number of communications and so to expand the system lifetime by conserving the energy of non-activated objects.

ACKNOWLEDGMENT

This research was funded by the Conseil Régional de Bourgogne-Franche Comté through the “plan d’actions regional pour l’innovation (PARI)” and the European Union through the “PO FEDER-FSE Bourgogne 2014/2020 programs”.

REFERENCES

[1] A. Khalil, N. Mbarek, and O. Togni, “QBAIoT: QoS Based Access for IoT Environments,” The Fourteenth Advanced International Conference on Telecommunications (AICT 2018), 2018, pp. 38–43, ISBN: 978-1-61208-650-7.

[2] A. Nordrum, “Popular IoT Forecast of 50 Billion Devices by 2020 Is Outdated”, IEEE Spectrum, August 2016.

[3] ITU-T Y.2066, “Next Generation Networks – Frameworks and functional architecture models”, 32 pages, 2014.

[4] IEEE Standard for Local and metropolitan area networks, Low-Rate Wireless Personal Area Networks, IEEE Computer Society, 311 pages, September 2011.

[5] P. Thubert, C. Bormann, L. Toutain, and R. Cragie, “Pv6 over Low-Power Wireless Personal Area Network (6LoWPAN) Routing Header”, IETF RFC, 37 pages, April 2017.

[6] S. Nath, S. Aznabi, N. Islam, A. Faridi, and W. Qarony, “Investigation and Performance Analysis of Some Implemented Features of the ZigBee Protocol and IEEE 802.15.4 Mac Specification”, International Journal of Online Engineering (iJOE), vol.13, pp. 14-32, Nov.2017, ISSN: 1861-2121, doi:10.3991/ijoe.v13i01.5984

[7] ITU-T Y.2060, “Y.2060: Overview of the IoT, ITU-T”, 22 pages, 2012.

[8] ISO/IEC JTC 1, “IoT (IoT) Preliminary Report 2014”, 17 pages, 2015.

[9] J. Jimenez, H. Tschofenig and D. Thaler, “Report from the IoT (IoT) Semantic Interoperability (IOTSI) Workshop 2016”, Internet Draft, 17 pages, July 2018.

[10] O. Garcia-Morchon, S. Kumar and M. Sethi, “State of the Art and Challenges for the IoT Security”, Internet Draft, 47 pages, December 2018.

[11] P. Mell and T. Grance, “The NIST Definition of Cloud Computing”, NIST, 2 pages. version 15, July 2009.

[12] A. Banafa, “Definition of fog computing”, IBM, August 2014, https://www.ibm.com/blogs/cloud-computing/2014/08/fog-computing/, (Last access 17 March 2018).

[13] International Electrotechnical Commission, “IEC role in the IoT”, 20 pages, 2017.

[14] World Health Organization, "WHO and PATH partner to globalize digital health", September 2018, https://www.who.int/ehealth/events/WHO-PATH-partnership/en/, (Last Access 14 January 2019).

[15] Ericsson, "Ericsson and partners demonstrate battery life improvements in Massive IoT e-health wearable prototype", September 2018, https://www.ericsson.com/en/news/2018/8/connected-e-health-IoT, (Last Access 14 January 2019).

[16] Nokia, " Enabling the human possibilities of smart cities", https://networks.nokia.com/smart-city, (Last Access 14 January 2019).

[17] 4G Americas, "Cellular Technologies Enabling the IoT", 2015, http://www.5gamericas.org/files/6014/4683/4670/4G_Americas_Cellular_Technologies_Enabling_the_IoT_White_Paper_-_November_2015.pdf, (Last Access 14 January 2019).

[18] Lora Alliance, "A Technical Overview of LoRa® and LoRaWAN™", https://www.tuv.com/media/corporate/products_1/electronic_components_and_lasers/TUeV_Rheinland_Overview_LoRa_and_LoRaWANtmp.pdf (Last Access 14 January 2019).

[19] ITU-T E.800, "Definitions of terms related to quality of service", 30 pages, 2008.

[20] J.Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "IoT (IoT): A vision, architectural elements, and future directions", Future Generation Computer Systems, vol. 29, pp. 1645-1660, 2013, doi: 10.1016/j.future.2013.01.010

[21] R. Bhaddurgatte and V. Kumar, "Review: QoS Architecture and Implementations in IoT Environment", Research & Reviews: Journal of Engineering and Technology, ISSN: 2319-9873, pp. 6-12, 2015.

[22] B. Ray, "Benefits of Quality of Service (QoS) in LPWAN for IoT", LinkLabs, December 2016

[23] M. Serrano, "OpenIoT D.4.6 Quality of Service (QoS) for IoT services", OpenIoT Consortium, Project Number 287305, 51 pages, 2014.

[24] M. A. Nef, L. Perleps, S. Karagiorgou, and G. I. Stamoulis, "Enabling QoS in the IoT", The Fifth International Conference on Communication Theory, Reliability, and Quality of Service, May 2012.

[25] S. Ezdiani, I. S. Acharyya, S. Sivakumar, and A. Al-Anbuky "An IoT Environment for WSN Adaptive QoS", 2015 IEEE International Conference on Data Science and Data Intensive Systems (DSDIS 2015), , 2015, pp. 586-593, ISBN: 978-1-5090-0214-6, doi:10.1109/DSDIS.2015.28

[26] S. Sarode and J. Bakal, "A Slotted CSMA/CA of IEEE 802.15.4 Wireless Sensor Networks: A Priority Approach", International Journal of Computer Trends and Technology (IJCTT), vol. 44, pp. 33-38, Feb. 2017, ISSN: 2231-2803, doi: 10.14445/22312803/IJCTT-V44P106

[27] F. Xia, J. Li, R. Hao, X. Kong, and R. Gao, "Service Differentiated and Adaptive CSMA/CA over IEEE 802.15.4 for Cyber-Physical Systems", The Scientific World Journal, vol. 2013, Article ID 947808, 12 pages, 2013, doi:10.1155/2013/947808

[28] A. Khalil, N. Mbarek, and O. Togni, "Service Level Guarantee Framework for IoT environments", International Conference on IoT and Machine Learning (IML 2017), 2017, ISBN: 978-1-4503-5243-7, doi: 10.1145/3109761.3158393

[29] M. Kirsche, IEEE 802.15.4-Standalone, https://github.com/michaelkirsche/IEEE802154INET-Standalone (Last Access 17 March 2018)