Proceedings of the 18th Conference on Computer Science and Intelligence Systems pp. 931–936
DOI: 10.15439/2023F8513
ISSN 2300-5963 ACSIS, Vol. 35

Real-time Communication Model for IoT Systems

Stanisław Deniziak
0000-0002-6812-5227
Kielce University of Technology
Faculty of Electrical Engineering, Automatic Control and Computer Science, Kielce, Poland
s.deniziak@tu.kielce.pl

Mirosław Płaza
0000-0001-9728-3630
Kielce University of Technology
Faculty of Electrical Engineering, Automatic Control and Computer Science, Kielce, Poland
m.plaza@tu.kielce.pl

Łukasz Arcab
0000-0003-4726-732X
Kielce University of Technology
Faculty of Electrical Engineering, Automatic Control and Computer Science, Kielce, Poland
lukasz.arcab@protonmail.com

***Abstract*—Internet of Things solutions typically involve interaction between sensors, actuators, the cloud, embedded systems and user applications. Often in such cases, there are time constraints specifying the maximum response time to a request. This time depends on the calculation time and transmission time. Existing Internet communication solutions do not ensure the implementation of transmissions in a way that guarantees meeting the set time constraints. This paper proposes a new model of Internet communication dedicated to real-time Internet of Things systems, which includes a communication protocol, as well as a transmission scheduling and routing method. The protocol takes into account information about transmission time constraints, which is used for packet scheduling by routers, allowing to increase quality of service. In addition, the proposed static routing mechanism makes it possible to parallelize transmissions if time constraints are still exceeded. Also presented are preliminary results of experiments showing to what extent the proposed methods allow improving the quality of service in real-time Internet of Things systems.**

***Index Terms*— real time routing, tasks scheduling, IoT, communication protocols.**

I. Introduction

The rapid development of the Internet of things (IoT) concept has led to a very large increase in interest in these solutions in almost all areas of our lives [1-3]. In the age of accelerating solutions in this area, there is a steadily increasing demand for IoT systems that, using various communication technologies, will also meet real-time requirements [4]. An important challenge of this research direction is to ensure that time requirements can be met optimally. The need to design real-time IoT (RTIoT) systems was recognized more than 10 years ago [5], when the first technologies and standards to support these solutions began to emerge (e.g.: Time Coordinated Computing (TCC) [6, 7], or the IEEE 802.1 standard – Time Sensitive Network (TSN) [8]). Usually, however, the solutions known today do not guarantee a satisfactory level of Quality of Service (QoS), which in most cases is crucial for the correct operation of the designed system. Therefore, it is necessary to undertake research work to develop methods and technologies to build IoT applications that meet real-time requirements. The first stage of this work was the development of the RTIoT system design methodology by the authors [9].

The key elements of the aforementioned methodology are to propose efficient scheduling and routing methods dedicated directly to RTIoT systems. The primary task of implementing such solutions will be to obtain better QoS performance values relative to standard scheduling methods, especially routing. In this case, the QoS value should be calculated for the worst case, i.e. for conditions that determine the maximum expected load on the system.

Thus, the research problem can be formulated as follows: given is a set of *N* endpoint devices and computing nodes of an IoT network that can send and/or receive transmitted data. The individual devices are interconnected using the network infrastructure that includes, among other things, va.rious communication links of a certain bandwidth and active devices, including routers. Given are also *M* different types of transmission between endpoint devices. In addition, there are strict time requirements associated with selected transmissions. Thus, it is necessary to find a solution for organizing the transmission in order to achieve the best QoS parameters (such as the average QoS of all real-time transmissions), that is, to minimize the average violation of time constraints. For this purpose, transmission scheduling algorithms (determining the order in which data is transmitted) as well as routing algorithms (optimizing routing for individual transmissions) can be used. This work assumes that the above problem will be optimized using static routing. This will ensure the predictability of the developed solutions, enabling the design of RTIoT systems based on small and medium-sized networks, such as a metropolitan area network.

The subsequent part of the article is organized as follows. Section II analyses the current state of the art of communication protocols currently used in IoT systems. Section III describes the assumptions that define the specification of the proposed system and the QoS optimization assumptions. Section IV proposes a transmission scheduling algorithm and an algorithm for selecting optimal routes. Section V contains the results of the experiments conducted, while Section VI presents conclusions and directions for further research.

II. Related work

In real-time IoT systems, the use of appropriate types of transmission media and the implementation of proper communication protocols, with particular emphasis on routing methods, plays an important role. Known communication protocols that are worth considering when designing RTIoT solutions include: RTSP (Real Time Streaming Protocol) [10], WebRTC (Web Real Time Communication), XMPP (Extensible Messaging and Presence Protocol) [11], MQTT (Message Queue Telemetry Transport) [12], CoAP (Constrained Application Protocol) [13], WebSocket [14], 6LoWPAN (IPv6 over Low-Power Wireless Personal Area Networks) [15].

The most well-known routing protocols used in IoT systems include:

- RPL (Routing Protocol for Low Power and Lossy Network – LLNs). LLNs are devices characterized by low power consumption, memory, and reduced resource engagement for processes. This protocol belongs to the family of distance vector protocols, which has been designed to work on multiple links [16].
- CTP (Collection Tree Protocol), which is a distance vector routing protocol and was developed for packet routing in WSNs (Wireless Sensor Networks). This protocol assumes the construction of a network topology tree taking into account routes for potential data packets [17].
- LOADng (Lightweight On-Demand Ad hoc Distance Vector Routing Protocol – next generation), is a lighter version of the AODV (Ad hoc On-Demand Distance Vector) protocol for LLNs. It was designed on the premise that LLNs are unoccupied for most of their time. This protocol follows an approach in which routes are determined in the direction of the packet's destination only when there is data to be sent [18].
- CORPL (Cognitive Radio RPL routing protocol), which is based on the RPL routing protocol and, with the modifications made, enables its use in Cognitive Radio environments [19].
- CARP (Channel-Aware Routing Protocol) – a protocol that uses a multi-hop approach to deliver data packets for WSNs. CARP has the advantage of taking link quality into account in the node selection process for next-hop [20].
- E-CARP (Enhanced CARP) – it is characterized by energy efficiency in the process of transmitting packets from the transmitter to the destination. In addition, this protocol does not differentiate the priority of attributes [21].

When it comes to designing real-time IoT networks, it is important to consider the issues of routing protocols and aim to achieve the best QoS transmission parameters. Based on their ability to deliver packets at specific/set deadline values, these protocols can be divided into two major groups: hard real-time and soft real-time [22]. Real-time routing protocols include:

- QoSR (Quality-of-Service Routing) – its greatest asset is its low energy consumption in determining the path for delivering data packets from source to destination. Unfortunately, the protocol exhibits poor support for scalable networks [23].
- QoSAM (QoS Aware Multi-Hop) – similarly to the QoSR protocol, it solves the problem of excessive energy consumption in determining the path for a packet from source to destination. Unfortunately, this protocol has much room for improvement in terms of reliability [24].
- MIMO (Multiple Inputs and Multiple Output) – a protocol that is dedicated to widely scaled WSNs. The implementation of this routing protocol offers the benefits of better energy utilization, lower transmission delays, packet loss and better bandwidth utilization of the transmission link [25].
- PRTR (Potential-based Real-Time Routing) – similarly to the MIMO routing protocol, it is characterized by scalability and a reduced probability of packet loss during transmission – resulting in the protocol requiring additional power, energy [26].
- QEMPAR (QoS and Energy Aware Multi-Path Routing Algorithm) – increases the lifetime of the network, unfortunately at the cost of increased delays [27].
- PT (Pheromone Termite) – features very good packet transmission performance by using a termite-based approach for routing. The protocol specifically focuses on finding the shortest route while maintaining QoS requirements. The PT protocol provides two new properties: pheromone sensitivity, which helps determine the link throughput, and packet generation rate, which helps update nodes in relation to the number of generated packets. One of the disadvantages of this protocol is that it is dedicated to large-scale networks [28].

A lack of an approach that takes into account proper packet scheduling (using appropriate scheduling methods) with deadline values, which can translate into improved QoS performance values, can be noticed in all of the above-mentioned communication protocols.

III. Assumptions

IoT systems usually have predefined functions, i.e. they can be specified in the form of a set of communicating tasks. In most cases, it is possible to estimate task execution times (e.g. for the worst case) and transmission volumes. Thus, for RTIoT systems, a design methodology analogous to that used for distributed embedded systems can be proposed [29].

According to our RTIoT system design methodology, the system specification is represented by a set of annotated task graphs (ATGs) [30]. Each ATG can be activated at a certain maximum frequency. The maximum number of instances of a given graph is also given. Designing an RTIoT system involves mapping the specifications to a target architecture consisting of 4 layers (Fig. 1): the Sensor Layer (SL), the Edge Layer (EL), the Cloud Layer (CL) and the User Layer (UL)

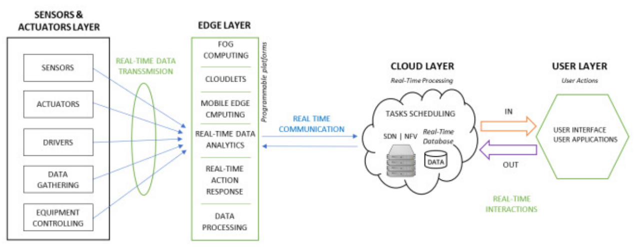

Fig. 1. General architecture of a real-time IoT system

A. System specification

A task graph is a directed acyclic graph $G=\{V,E\}$ in which nodes $v_i \in V$ represent tasks and edges $e_{i,j} \in E$ describe relationships between tasks, usually related to communication. The attributes of the graph describe the assignment of tasks to the layers of the RTIoT architecture and transmission volumes between tasks. Sample annotated task graph is shown in Fig. 2. Attributes that define layers are represented by colours. Attributes describing transmissions are represented by edge labels. From the perspective of network communication, only the transmissions between layers are relevant.

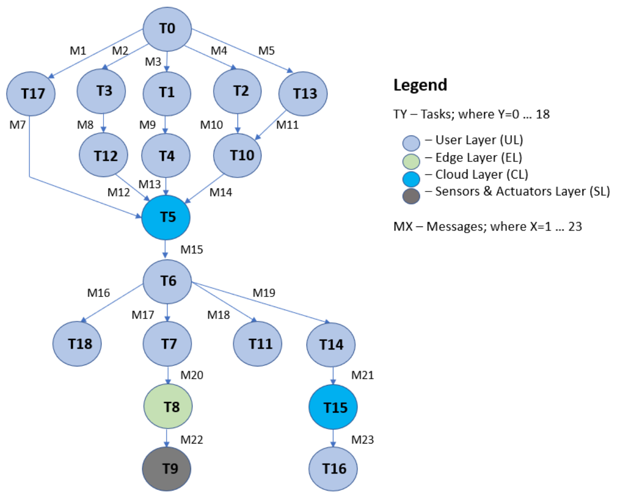

Fig. 2. Sample annotated task graph

Graph given in Fig. 2 describes the main function of a smart city system for managing parking spaces [9]. The user activated task graph specifies the following system functionalities: searching for the parking space closest to the user's current location, the function of finding the user's car in the parking lot based on their license plate number, the function of reserving any free parking space based on the entered search criteria, the function of charging a parking fee for the used parking lot, and the function of retrieving information on weather conditions.

In the example presented in Fig. 2, one task graph activation can cause the following transmissions: M7, M12, M13, M14, M15, M20, M21, M22, M23, where successive numbers indicate individual transmission types. Multiple instances of a given task graph can be activated at any given time, caused by the occurrence of multiple simultaneous events activating given function. For example, multiple users can run an application that sends requests to an IoT system. Thus, there may be a large number of simultaneous transmissions causing a significant load on communication links, leading to the violation of the time constraints.

B. QoS optimisation

A d_{max} time constraint can be associated with any v_i task. This constraint determines the time in which the task should be completed. Soft real-time systems are considered in the paper. In such systems, two types of restrictions are defined: d^h_{max} and d^s_{max}, The d^s_{max} constraint can be exceeded but then the quality of service (QoS) is lower. The d^h_{max} constraint specifies the maximum time that can no longer be exceeded. Violation of the d^h_{max} constraint means packet lost.

The goal of optimizing RTIoT systems is to achieve the highest possible QoS. QoS for a single constraint can be defined as:

$$QoS_i = \begin{cases} 0 \text{ when } t_i > d^h_{max} \\ 1 \text{ when } t_i < d^s_{max} \\ 1 - \frac{t_i - d^s_{max}}{d^h_{max} - d^s_{max}} \text{ in other cases} \end{cases} \tag{1}$$

where: t_i – is the current finish time of task covered by the i-th constraint.

Then the total QoS for the system can be determined as the average value of all QoS_i:

$$QoS = \sum_{i=0}^{n} \frac{QoS_i}{n} \tag{2}$$

where: n – is the number of all constraints in all instances of task graphs.

If for task v_i constraints d^h_{max} and d^s_{max} are specified then time constraints for all transmissions represented by edges $e_{x,i}$ entering node v_i can also be specified as follows:

$$\text{ex}d^h_{max} = d^h_{max} - te_i \tag{3}$$

$$\text{ex}d^s_{max} = d^s_{max} - te_i \tag{4}$$

where: te_i – is the expected execution time of task v_i, usually determined by WCET (Worst Case Execution Time) estimation.

Time constraints for all transmissions can be determined in an analogous way. Thus, the goal of transmission optimization will be to organize the transmission of messages in such a way that each transmission ends before $\text{ex}d^s_{max}$ or exceeds this time as little as possible while not exceeding $\text{ex}d^h_{max}$. This can be achieved by appropriate transmissions scheduling and/or the use of routing that minimizes collisions of simultaneous transmissions.

IV. Real-time Routing

Existing methods of Internet communication are mainly based on ensuring the most efficient transmission. They do not take into account time constraints or the issue of predictability of transmission time. For these reasons, these methods are not suitable for RTIoT systems.

In order to take into account time constraints, the routing method should use transmission scheduling mechanisms in such a way as to minimize delays and not use overly time-consuming route determination algorithms.

A. Transmission scheduling

It is assumed that individual packets contain information identifying real-time transmissions. Real-time transmissions are processed in the first step, in the order determined by the scheduling algorithm. The remaining transmissions are processed in FIFO order when the list of real-time transmissions is empty. The real-time transmission scheduling algorithm is based on the Least Laxity First (LLF) algorithm [31], which provides optimal task scheduling in real-time systems. Associated with each such packet is information specifying the deadline exd^s_{max} and the expected transmission time tt_i estimated based on the length of transmission and the average bandwidth of communication links. Draft scheduling algorithm is shown in Fig. 3.

```
Schedule (e_i, RTList){
if RTList[0]=Φ
    TList[0]=e_i
else {
    pos=0;
    while RTlist[pos]!= Φ
       if (Laxity(Tlist[pos])<Laxity(e_i)) pos++;
       else {
          Insert(e_i, RTList, pos);
          break;
       }
  }
return RTList;
}
```

Fig. 3. Draft transmission scheduling algorithm

The *Laxity(e_i)* function computes the transmission time reserve as follows:

$$L_i = exd^s_{max} - tt_i \quad (5)$$

The *Insert(ei, RTList, pos)* function inserts the transmission e_i into the *RTList* at the *pos* position. Thus, the scheduling algorithm creates a list of transmissions ordered from the smallest value of L_i.

B. Choice of routes

The choice of transmission routes for transmitting individual packets affects both transmission time and collision-related delays. Thus, the main goals of optimization should be to find the shortest routes and avoid collisions for simultaneous transmissions. Collisions cannot be avoided for transmissions using the same transceivers or receivers.

In the case of real-time systems, it is crucial to implement the transmission such that the violation of time constraints is eliminated or minimized. In addition, the real-time system should be predictable, only then can adequate QoS be guaranteed under a set system load. Predictability can only be achieved with static routing. Then, assuming that the network topology is fixed and the network load is known, the assumed minimum QoS level will be guaranteed. We also assume that all non-colliding paths between given nodes may be found using existing methods e.g. as in the NoC systems [30].

The problem of optimizing real-time transmission for a given network topology can be defined as the problem of allocating communication routes for worst-case scenarios. Suppose that at any given time, *m* transmissions of data M_1, ..., M_m need to be made between S_i and D_i nodes. Then, if after scheduling the transmissions according to the algorithm from Fig. 3, the transmission delay, for any transmission, resulting the position in the list will cause the deadline to be exceeded, it means that it is necessary to send packets through different routes in order to parallelize the transmissions. Otherwise, all packets can be sent via a single route.

The algorithm for allocating transmissions to routes is shown in Fig. 4. The input to the algorithm is a list of non-colliding *PList* routes and an ordered list of *RTList* transmissions. The algorithm then sequentially schedules transmissions for the next paths in a loop. Transmissions allocated to routes are removed from the *RTList*. The *Time* counter adds up the times of consecutive transmissions allocated to a given path. If the allocation of the next transmission to a particular path results in exceeding the deadline for that transmission then the transmission remains in the list and the algorithm will try to allocate it to the next path in the next loop run. The algorithm returns the number of routes required to complete all transmissions, or an ERROR value if it fails to ensure that all transmissions complete within the required time. In that case, either the network topology needs to be modified to create more routes, or the remaining ones need to be allocated with a minimal violation of the time constraint in order to achieve the lowest QoS drop.

```
AssignPath(Plist, RTList) {
  PathNo=0;
  do {
       Path=Plist[PathNo];
       Time=0;
       For (Pos=0; Length(RTList); Pos++)
        if (Laxity[RTList[Pos]-Time >=0) {
          Assign(RTList[Pos], Path);
          Time+=RTList[Pos].tt;
          Remove(RTList[Pos]);
       }
       if (RTList==Φ) return PathNo;
       PathNo++;
    }
    while PList[PathNo]<>Φ;
    return ERROR;
}
```

Fig. 4. Draft route assignment algorithm

When transmissions involve different destination nodes, routes and ordered *RTLists* should be determined for each node and the algorithm shown in Fig. 4 should be performed independently for each pair of lists *(RTList*, *PList*).

V. Experimental results

The work performed included four experiments. Each of them was performed with given initial conditions such as: equal bandwidth of transmission links; no other type of data packets in the network; 100 different packet transmissions were assumed in the same period, with transmission time for a single packet not exceeding 50ms. Soft deadline (ranging from 1000ms to 5000ms) and hard deadline (ranging from

2000ms to 7000ms) values were also set randomly. The experiments may correspond to any transmission from Fig. 2, between 2 layers (e.g. M20), assuming simultaneous activation of this transmission by 100 users.

The first experiment was conducted for a network in which communication over a single transmission link is assumed. Packet handling by a router with an implemented static routing mechanism is done according to a random packet queue. With this type of approach, it is observed that QoS requirements are not met for a lot of transmitted packets. For 40% of all transmissions the QoS were lower than 1 and 13% of transmissions failed i.e. the hard deadlines were not fulfilled.

The second experiment is an extension of the first approach, which was extended to include the implementation of the LLF-based packet scheduling algorithm (described in Section IV.A of this paper). The results of this experiment clearly show the benefits of using packet scheduling. With appropriate transmission scheduling using the LLF-based algorithm, transmission quality improvement is achieved by obtaining better QoS parameter values with respect to the original values. Only 9% of transmissions exceeded the soft deadline. Thus, this approach is closer to meeting the conditions for real-time transmission.

The third experiment assumed the existence of two independent routes through which packets can be sent using routing mechanisms. In addition, for the purposes of the experiment, it was assumed that the distribution of packets between the previously mentioned routes is even, i.e. half of the previously assumed 100 packets are routed through one link and the remainder through the other link. The results of this experiment showed that, despite the existence of a second, alternative communication link, not all individual transmissions were able to achieve satisfactory QoS results – not all packets (only 87%) were delivered while maintaining the QoS parameter at the level specified by the soft deadline.

The last experiment is an extension of the approach tested in the third experiment. In this case, as in the second experiment, the LLF algorithm that schedules data packets was used. The results of this experiment showed that the existence of two routes in combination with the implementation of a packet data scheduling algorithm allows for the best results in terms of QoS parameters. In this case, all individual transmissions achieved the highest value of the QoS parameter equal to 1. Table I presents a summary of the results obtained for all four experiments conducted. The first column (PAR) defines the parameter name. The following rows contains values of: Average transmission time (ATT), soft deadlines (SD), hard deadlines (HD), number of messages (NM), the number of transmissions that exceeded the soft deadline (NM<SD), the number of transmissions that exceeded the hard deadline (NM<HD), Quality of Service (QoS) obtained for each experiment.

Fig. 5-8 illustrate the dependence of subsequent data transmissions on QoS parameters. Transmissions that did not meet any QoS requirements in the experiments were marked in red, transmissions that only met the requirements of the soft deadline were marked in blue, and those that met all requirements were marked in green.

Analysing the results of the research, it can be seen that for the first experiment, 27 different transmissions did not meet the requirements of the soft deadline, while 13 did not meet the requirements of the hard deadline. In the second and third experiments 9 and 12 different transmissions, respectively, did not meet the requirements of the soft deadline. In the last experiment, all QoS requirements for all types of transmissions were met.

TABLE I. SUMMARY OF EKSPERIMENTAL RESULTS

PAR	Random Singe route	LLF Singe route	Random Double route	LLF Double route
ATT	3434ms			
SD	1000ms, 3000ms, 5000ms			
HD	2000ms, 5000ms, 7000ms			
NM	100			
NM < SD	27	9	13	0
NM < HD	13	0	0	0
QoS	**0,81**	**0,97**	**0,95**	**1**

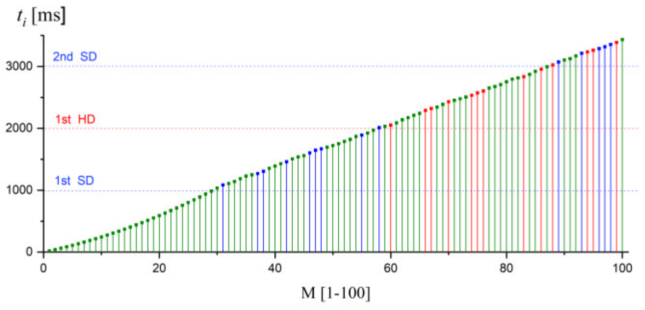

Fig. 5. Dependence of QoS parameters on subsequent data transmissions for first experiment

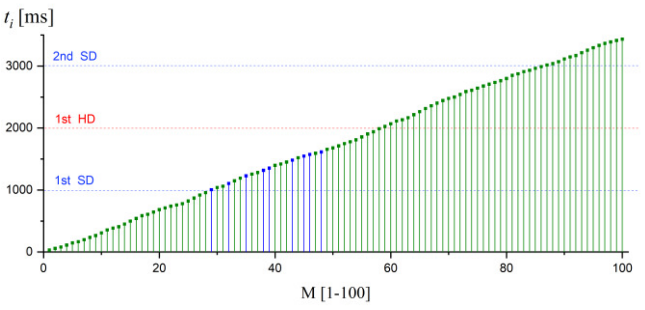

Fig. 6. Dependence of QoS parameters on subsequent data transmissions for second experiment

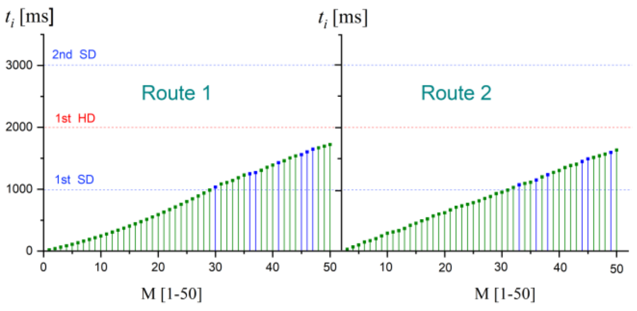

Fig. 7. Dependence of QoS parameters on subsequent data transmissions for third experiment a) first route, b) second route

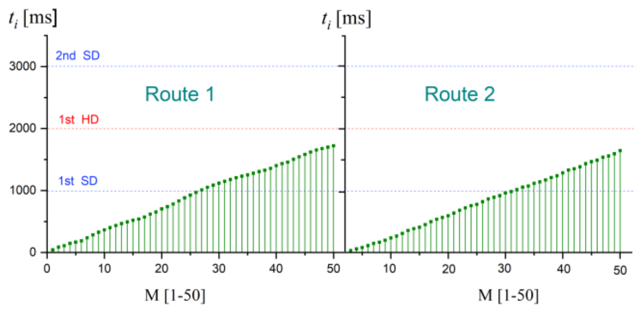

Fig. 8. Dependence of QoS parameters on subsequent data transmissions for fourth experiment a) first route, b) second route

VI. CONCLUSIONS

The article presents the model of Internet communication, dedicated directly to the needs of IoT systems, where real-time requirements are particularly important. The proposed solution is based on data transmission scheduling algorithms and the use of routing methods. The model takes into account information about time constraints at both the soft deadline level and the hard deadline level. Both proper data scheduling and routing mechanisms improve QoS parameters in the system under consideration, as demonstrated by the experiments presented in the paper.

The experiments, conclusions and observations that follow indicate the justification of the approach in which both packet data scheduling methods and appropriate routing methods are applied in RTIoT networks. Based on the simulations and calculations, it should also be noted that the number of routes used for packet transmission also plays an important role in improving QoS parameters for both individual data transmissions and the entire designed system.

The future work on the presented topic will focus on further improvements and extensions to the discussed model. In particular, we will address the implementation capabilities of dynamic routing protocols, as well as other known scheduling methods. The result will be a complete RTIoT system design and implementation environment, ensuring the development of systems with a high level of QoS.

REFERENCES

[1] M. Płaza, R. Belka, Z. Szcześniak, "Towards a different world – on the potential of the Internet of everything", IAPWGIOS, vol. 9(2), pp. 8-11, June 2019, https://doi.org/10.5604/01.3001.0013.2539"

[2] P. Pięta, S. Deniziak, R. Belka, M. Płaza, and M. Płaza, "Multi-domain model for simulating smart IoT-based theme parks", Proc. SPIE 10808, Photonics Applications in Astronomy, Communications, Industry, and High-Energy Physics Experiments 2018, 108082T, October 2018.

[3] R. Belka, S. Deniziak, M. Płaza, M. Hejduk, P. Pięta, M. Płaza, P. Czekaj, P. Wołowiec, K. Ludwinek, "Integrated visitor support system for tourism industry based on IoT technologies", Proc. SPIE 10808, Photonics Applications in Astronomy, Communications, Industry, and High-Energy Physics Experiments 2018, 108081J, October 2018.

[4] M. Płaza, R. Belka, M. Płaza, S. Deniziak, P. Pięta, Sz. Doszczeczko, "Analysis of feasibility and capabilities of RTLS systems in tourism industry", Proc. SPIE, 10808, Photonics Applications in Astronomy, Communications, Industry, and High-Energy Physics Experiments 2018, 108080C, October 2018.

[5] S. Bąk, R. Czarnecki, S. Deniziak, "Synthesis of real-time cloud applications for Internet of Things", Turk. J. Elec. Eng. & Comp. Sci., vol 23, pp. 913-929, 2015, https://doi.org/10.3906/elk-1302-178

[6] Intel. Intel Time Coordinated Computing Tools. 2022. Available online: https://www.intel.com.

[7] Intel. Real-Time at the Edge: Overview. https://www.intel.com, 2022.

[8] J. Lee. S. Park, "Time-sensitive network (TSN) experiment in sensor-based integrated environment for autonomous driving", Sensors, vol. 19(5), pp. 1111, March 2019, https://doi.org/10.3390/s19051111

[9] S. Deniziak, M. Płaza, Ł. Arcab, "Approach for designing real-time IoT systems. Electronics, vol. 11(24), pp. 1-21, December 2022.

[10] J. Lee, J. Kim, S. Kim, Ch. Lim, J. Jung, "Enhanced distributed streaming system based on RTP/RTSP in resurgent ability", Proc. Fourth Annual ACIS ICIS'05, Jeju Island, South Korea, 14-16 July 2005.

[11] M. Kirsche, R. Klauck, "Unify to bridge gaps: Bringing XMPP into the Internet of Things", 2012 IEEE Int. Conf. on Pervasive Computing and Communications Workshops, Lugano, Switzerland, 19-23 March 2012

[12] MQTT. MQTT: The Standard for IoT Messaging. 2022. Available online: https://mqtt.org (accessed on 6 January 2023)

[13] C. Bormann, A.P. Castellani, Z. Shelby, "CoAP: An application protocol for billions of tiny internet nodes", IEEE Internet Computing, vol. 16, pp. 62 - 67, 2012.

[14] G. L. Muller, HTML5 WebSocket protocol and its application to distributed computing. https://arxiv.org/abs/1409.3367.

[15] M. Ha, D. Kim, S. H. Kim, S. Hong, "Inter-MARIO: A fast and seamless mobility protocol to support inter-pan handover in 6LoWPAN" 2010 IEEE GLOBECOM, 06-10 December 2010, pp. 1-6

[16] J.V.V. Sobral, J.J.P.C. Rodrigues, R.A.L. Rabêlo, J. Al-Muhtadi, V. Korotaev, "Routing Protocols for Low Power and Lossy Networks in Internet of Things applications", Sensors, vol. 19, pp. 2144, 2019.

[17] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, P. Levis, "The collection tree protocol (CTP)", Proc.(SenSys, November 2009

[18] T. Clausen, J. Yi, U. Herberg, "Lightweight on-demand ad hoc distance-vector routing - next generation (LOADng): Protocol, extension, and applicability", Computer Networks, vo. 126, pp. 125-140, 2017.

[19] Z. Yang, S. Ping, H. Sun, A. H. Aghvami, "CRB-RPL: A receiver-based routing protocol for communications in cognitive radio enabled smart grid", IEEE Trans. on Veh. Tech., vol.66(7), pp.5985-5994, 2017.

[20] S. Basagni, C. Petrioli, R. Petroccia, D. Spaccini, "Channel-aware routing for underwater wireless networks", Proc. Oceans-Yeosu, 2012.

[21] Z. Zhou, B. Yao, R. Xing, L. Shu, S. Bu, "E-CARP: An energy efficient routing protocol for UWSNs in the Internet of underwater things", IEEE Sensors Journal, vol. 16(11), pp. 4072-4082, June 2016.

[22] S. Malik, S. Ahmad, I. Ullah, D. H. Park, D. H. Kim, "An adaptive emergency first intelligent scheduling algorithm for efficient task management and scheduling in hybrid of hard real-time and soft real-time embedded IoT systems", Sustainability, vol. 11(8), pp. 2192, 2019

[23] A. M. Alkahtani, M. E.Woodward, K. Al-Begain, "An overview of Quality of Service (QoS) and QoS Routing in communication networks", Computer Science, 2003.

[24] A. Alanazi, K. Elleithy, "Real-time QoS routing protocols in wireless multimedia sensor networks: Study and analysis", Sensors, vol. 15, pp. 22209-22233, August 2015, https://doi.org/10.3390/s150922209

[25] N. Kumar R. Khanna, "A compact multi-band multi-input multi-output antenna for 4G/5G and IoT devices using theory of characteristic modes", Int. J. of RF and Microwave Comp.-Aided Eng., vol. 30(6), Jan. 2020.

[26] Y. Xu, F. Ren, T. He, C. Lin, C. Chen, S. K. Das, "Real-time routing in wireless sensor networks: A potential field approach", ACM Transactions on Sensor Networks, vol. 9(3), May 2013.

[27] S. R. Heikalabad, H. Rasouli, F. Nematy, N. Rahmani, „QEMPAR: QoS and Energy Aware Multi-Path Routing Algorithm for Real-Time Applications in Wireless Sensor Networks", International Journal of Computer Science Issues, vol. 8(1), pp. 466-471, January 2011.

[28] A. Razaque, K. Elleithy, "Pheromone termite (PT) model to provide robust routing over Wireless Sensor Networks", Proc. of the 2014 ASEE Zone 1, pp. 1-6, 2014.

[29] S. Deniziak, R. Tomaszewski, "Codesign of energy and resource efficient contention-free Network-on Chip for real-time embedded systems, 2018 11th NoCArc, Fukuoka, Japan, pp. 1-6, 2018.

[30] S. Deniziak, R.Tomaszewski, "Co-synthesis of contention-free energy-efficient NOC-based real time embedded systems", Journal of Systems Architecture, vol. 98, pp. 92-101, 2019.

[31] S. Teng, W. Zhang, H. Zhu, X. Fu, J. Su, B. Cui, "A Least-Laxity-First scheduling algorithm of variable time slice for periodic tasks", in Y. Wang (Ed.), Breakthroughs in Software Science and Computational Intelligence, IGI Global, pp. 316-333.