ISSN: 2229-6948(ONLINE)
DOI: 10.21917/ijct.2014.0132

COAP BASED ACUTE PARKING LOT MONITORING SYSTEM USING SENSOR NETWORKS

R. Aarthi[1] and A. Pravin Renold[2]
TIFAC-CORE in Pervasive Computing Technologies, Velammal Engineering College, India
E-mail: [1]aardolphin@gmail.com, [2]pravinrenold.tifac@velammal.edu.in

Abstract

Vehicle parking is the act of temporarily maneuvering a vehicle in to a certain location. To deal with parking monitoring system issue such as traffic, this paper proposes a vision of improvements in monitoring the vehicles in parking lots based on sensor networks. Most of the existing paper deals with that of the automated parking which is of cluster based and each has its own overheads like high power, less energy efficiency, incompatible size of lots, space. The novel idea in this work is usage of CoAP (Constrained Application Protocol) which is recently created by IETF (draft-ietf-core-coap-18, June 28, 2013), CoRE group to develop RESTful application layer protocol for communications within embedded wireless networks. This paper deals with the enhanced CoAP protocol using multi hop flat topology, which makes the acuters feel soothe towards parking vehicles. We aim to minimize the time consumed for finding free parking lot as well as increase the energy efficiency.

Keywords:

Acute Parking Lots, CoAP, Internet of Things, Sensor Networks, Vehicle Monitoring

1. INTRODUCTION

Entering a new era of computing technology, called as the Internet of Things (IoT) which is also known as Machine to Machine, Machine to Infrastructure, Machine to Environment, Internet of Everything, Internet of Intelligent Things, Intelligent Systems [1]. The Internet of Thing is the network of physical objects accessed through the internet, as defined by technology analysts and visionaries. These objects contain embedded technology to interact the internal states or the external environment. In other words, when objects can sense and communicate, it changes how and where decisions are made and who makes them. Smart parking management systems are a part of the smart transportation systems of the future. Better management of parking facilities will get cars off the street and into parking spaces sooner thus contributing to congestion control in highly congested urban areas. A smart parking management system can help alleviate some of the pain by more efficiently using existing parking spaces and by making the process of finding a parking space quicker and less frustrating [2].

To deal with parking monitoring system issue such as traffic, this paper proposes a vision of improvements in monitoring the vehicles in parking lots based on sensor networks. Most of the paper [3]-[5] deals with that of the automated parking which is of cluster based and each has its own overheads like high power, less energy efficiency, incompatible size of lots, space. Car-park management systems operate by monitoring the availability of car-parking spaces and making that information available to customers and facility administrators. Customers use it for guiding them in their choice of parking space; administrators use it to aid in overall management and planning. Sensor networks are a natural candidate for car-park management systems [6], because they allow status to be monitored very accurately - for each parking space, if desired.

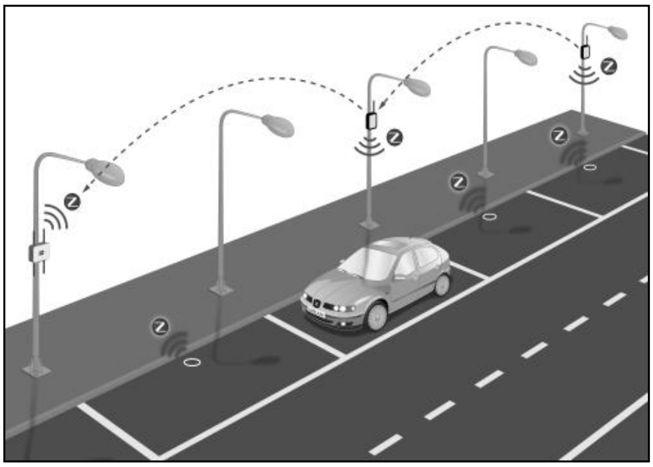

Fig.1. Car Parking Lot [7]

The Fig.1 shows the signal transmission between the transmitters and receivers using sensors in road side parking lots. Wireless sensor networks can be used for car-parking system as it have the advantage of deployment in existing car-parks without installing new cables for network and electricity to reach each sensing device. The analysis of variations in wireless sensor networks for car parking applications are tabulated in [8].

The goal of this work is to,

- Establish a web address for car parking lot via Copper Web Browser.
- User can log onto the web address and enter his/her car number and parking duration via his/her mobile devices.
- Sensors in the parking lot give the information regarding the availability of the car in the slots of the particular parking lot.
- According to the duration of the drivers need, slots are allocated.
- As per the request by driver, free parking slot will be displayed in his/her mobile devices, thus reducing the time spent for finding free space.

The contributions in this paper are: (1) a mathematical model for parking lot occupancy (2) a simulative evaluation of the proposed approach. The remainder of this paper is organized as follows. The next section discusses works related to car parking lots. Section 3 describes the proposed work of the paper and section 4 shows the evaluation methodology that is protocol to be used along with the simulation environment. The results are presented and discussed in section 5. Section 6 concludes the paper.

2. RELATED WORKS

Internet of Things has a great dimension in creating and maintaining the Smart Objects, Smart Environments. Our focus is on Smart parking applications.

Car parking application uses protocol such as LEACH (Low Energy Adaptive Clustering Hierarchy) and TEAN (Threshold sensitive Energy Efficient sensor Network) [5]. By using such protocols, Consumption of energy as well as traffic towards the sink is greater. In order to overcome such issues CoAP, which is an emerging open application layer protocol designed to support resource constrained machine-to-machine application environment [9] is used. The main challenge in CoAP is interoperability, which is held in that of application layer [10].

CoAP protocol was deployed in mobile environmental monitoring system for transmission of a resource description and sensor environmental data from IoT nodes and vehicle tracking devices [11].Therefore this protocol best suit for car parking applications. To overcome Interoperability, Web services have demonstrated to be essential in enabling interoperable communications between computers on the traditional Internet [12]. CoAP is a re-design from scratch following the REST architectural style [13]. It closes gap between microcontroller-based low-power devices and the web of things as HTTP over TCP is not feasible for constrained environments. Restful applications use URIs for addressing and uniform interfaces for interaction. Therefore URI (Uniform Resource Identifier) is created for accessing the application via browser. Thus CoAP have been implemented for various applications and this paper highlights CoAP for Car parking Applications.

3. OVERVIEW OF CoAP

Constrained Application Protocol (CoAP) is a specialized web transfer protocol which suits for constrained nodes as well as for unconstrained networks [14].

CoAP URI is made for this work is coap: //vec:6000

The method definitions involved in constrained application protocol are GET, POST, PUT, and DELETE. Message types involved are Confirmable, Non confirmable, Acknowledgement, Reset messages. Confirmable requires Acknowledgement whereas Non Confirmable doesn’t require Acknowledgment. Reset message indicates missing of few contexts.

Table.1. Method Definitions

METHODS	SAFE	IDEMPOTENT	DESCRIPTION
GET	YES	YES	Retrieves data
POST	NO	NO	Provides data
PUT	NO	YES	Updates data
DELETE	NO	YES	Removes data

The Table.1 shows the detailed description of the method definitions. The methods included are,

- GET-The GET method helps in retrieving information from that of server.
- POST-The POST method requests the server to provide data for the user.
- PUT-The PUT method helps in updating or creating the information in the corresponding URI.
- DELETE - The DELETE method helps in erasing the information stored.

There are two responses in CoAP namely

- Piggy-backed Response
- Separate Response

A Piggy-backed Response is to acknowledge whereas Separate Response is sent by an Empty message.

The usage of CoAP in car parking application is to form a client-server relationship, whereas drivers use the browser as client and the server program runs in the GUI for updating the browser dynamically. Thus the Method definitions are used efficiently for parking application.

4. PROPOSED ARCHITECTURE FOR CAR PARKING APPLICATION

4.1 SYSTEM ARCHITECTURE

The Fig.2 shows the system architecture which comprises of Car parking lot, Light Emitting Diodes (LED), mobile devices with internet connectivity. The free parking slots are intimated to the user via web address as per request. The data transmission from internet to that of user uses a protocol termed as Constrained Application Protocol.

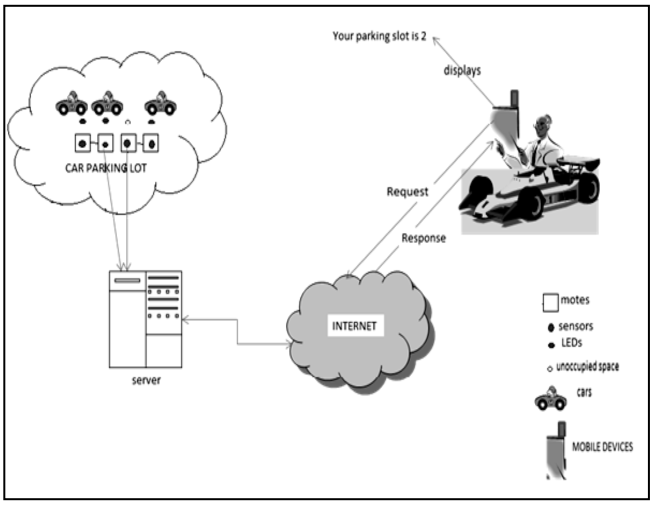

Fig.2. Architecture of Proposed System

As per Fig.3 the nodes used for parking application can be classified in to three categories such as one node acts as Server, one as Border Router and Clients may be in any number.

- Server - A Restful server shows how to use the REST layer to develop server-side applications.
- Border Router - Border Router keeps radio turned on. Enabling of it helps in connection between that of client

as well as server to that of CoAP web Address. Border-Router has the same stack and fits into mote memory.

- Client - A CoAP client cycles through 4 resources on event detection such as GET, PUT, PUSH, and DELETE. It gets connected to the server via multihop topology.

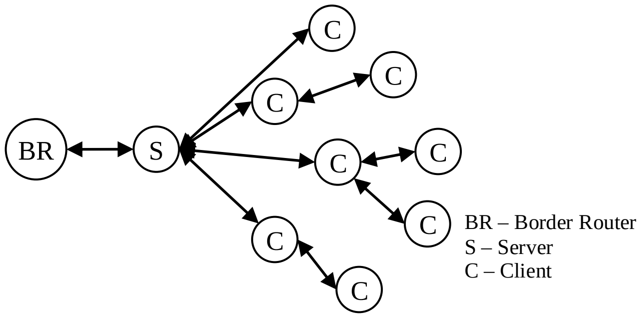

Fig.3. Communications between Border Router, Client and Server

The communication between server and clients are of multi hop fashion. The data from the motes are sent to that of Copper Web Browser [15] which can be viewed by the user.

4.2 MATHEMATICAL MODEL

Parking lot is considered to be a SIRO (Service In Random Order) queue and Markov chain is used to explain its memory less property. As vehicles can rest at a particular lot for few hours (i.e. temporary), a continuous-time model is used [16] [17]. For the queue representing the parking lot, homogeneous Markov model with exponentially distributed interarrival and parking times are used. The following equation shows the characteristic of non dependence of the past.

$$p_{ij}(t) := P(X_{\tau+t} = j \mid X_\tau = i) = P(X_t = j \mid X_0 = i) \quad (1)$$

whereas $P(X_{\tau+t} = j \mid X_\tau = i)$ represents the future occurrence and $P(X_t = j \mid X_0 = i)$ represents the present occurrence. Therefore the independence of the past occurrence as shown in Eq.(1).

Markov <=> Memory less.

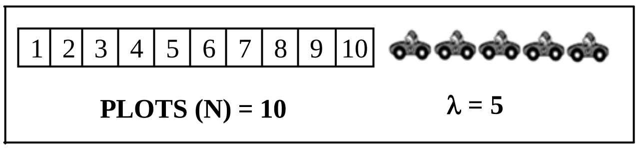

Fig.4. Car parking based on Parking Time

The car parking lot with 10 slots are filled as per the parking time is shown in Fig.4. Since the space on each parking lot is finite, the queue has the same capacity as the number of parking places on the parking lot [18]. Each state of the Markov model represents the respective number of vehicles currently parking on the parking lot. The mathematical model for the car parking application can be done via

$$\text{M/M/1: N/SIRO (TIME).} \quad (2)$$

The Eq.(2) shows that M is Markovian/Memory less (Poisson Distribution), M is Markovian/Memory less (Exponential Distribution), 1 is the Server and N is Number of Slots (Taken as 10), λ is cars arriving per hour = 5 and μ is the service / parking time (As $\mu \neq \lambda$) μ varies from 1 to 5 and from 6 to 10 hours. SIRO (TIME) says the discipline used in this paper as per the parking time of the driver. The waiting time is calculated by,

$$W_s = \frac{L_s}{\lambda'} \quad (3)$$

The Eq.(3) shows that W_s is the expected waiting time of the system, L_s deals with the expected number of cars and λ' deals with that of the effective length of the system. The concept used in filling the slots of parking lots are detailed in the below flowchart.

For $n : N$ (say 1 to 10)

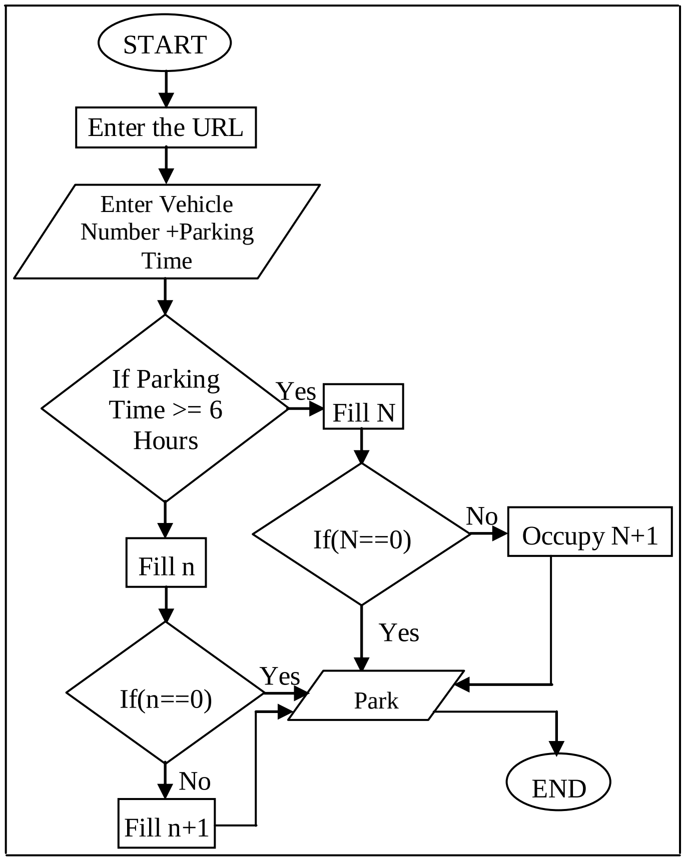

Fig.5. Flowchart based on Parking Time Algorithm

The Fig.5 details the proposed algorithm. It states that the slots are filled from N, N-1, N-2…for drivers who needs parking time more than 6 hours and starts filling from n, n+1, n+2 for parking time less than 5 hours.

Table.2. Results of existing Vs Proposed disciplines

	M/M/1:N/FIFO	M/M/1:N/SIRO (TIME)
W_s	<= 10 minutes	3 minutes (approximately)

The inference of the proposed algorithm is the waiting time for a car to park takes 3 minutes approximately which was proven by this mathematical model. The output is compared with that of algorithm which is based on random filling of slots in a parking lot regardless of the parking time is shown in Table.2.

5. SIMULATION RESULTS

Constrained Application Protocol is simulated via cooja simulator [19]. Cooja simulator is a java based simulator and it is also called as "cross level simulator".

The simulation can be done via the terminal as follows,

1. Server only: make TARGET=cooja server-only.csc

   Open new terminal:make connect-router-cooja is shown in Fig.6.

   - Start Copper and discover resources at coap://cooja2:5683/
   - Choose "Click button on Sky 2" from the context menu of mote 2 (server) after requesting /test/separate
   - Do the same when observing /test/event

2. With client: make TARGET=cooja server-client.csc

   Open new terminal: make connect-router-cooja

   - Wait until red LED toggles on mote 2 (server)
   - Choose "Click button on Sky 3" from the context menu of mote 3 (client) and Watch serial output.

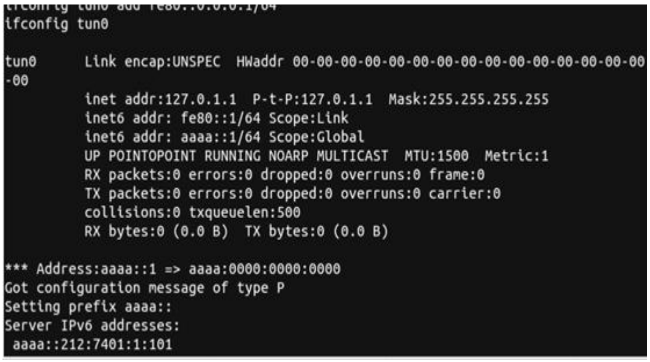

Fig.6. Connection of server-client via router

In CoAP protocol, Erbium CoAP is implemented (in which block wise transfer is possible, compatible with both server and client).The nodes in it are of three types namely CoAP Client, CoAP server, Border Router as shown in Fig.7.

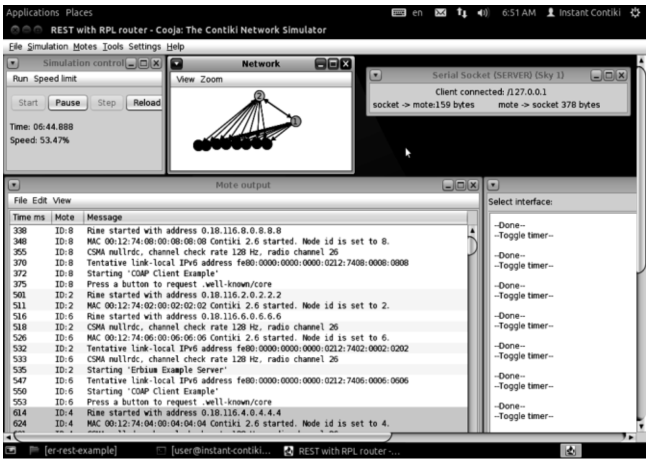

Fig.7. CoAP in Cooja Simulator-After Connection

Finally output can be viewed in Browser with a Mozilla Firefox add-on called Copper. Car is a resource added to the left side window of the browser in the address coap: //vec:6000/car and the driver can access the same link for knowing the status of the free parking lots.

The Fig.8 shows the process done in Mozilla Firefox browser

- Discovery – calls the resources included in the server.
- Ping – which helps to view the connectivity of the server and client.
- GET, POST, PUT, DELETE are the Method definitions which gets in to effect only after Pinging.

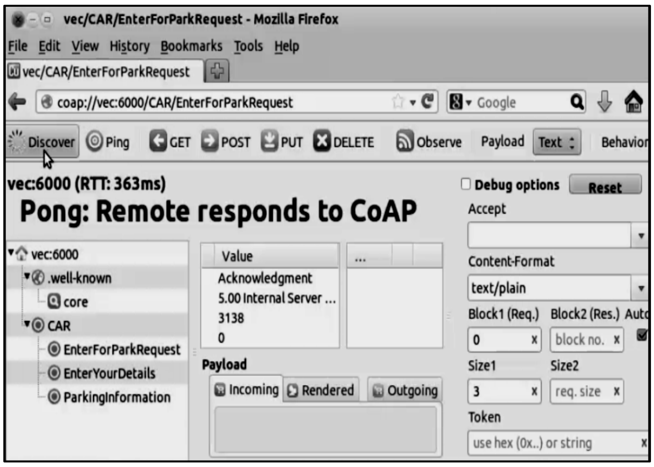

Fig.8. Pings in Copper CoAP Browser

The added sub-resources under car are EnterForParkRequest, EnterYourDetails and ParkingInformation.

- EnterForParkRequest says the user whether he/she can proceed for the parking procedure.
- EnterYourDetails asks for the details about vehicle number as well as the parking time.
- ParkingInformation gives the user about the free slots.

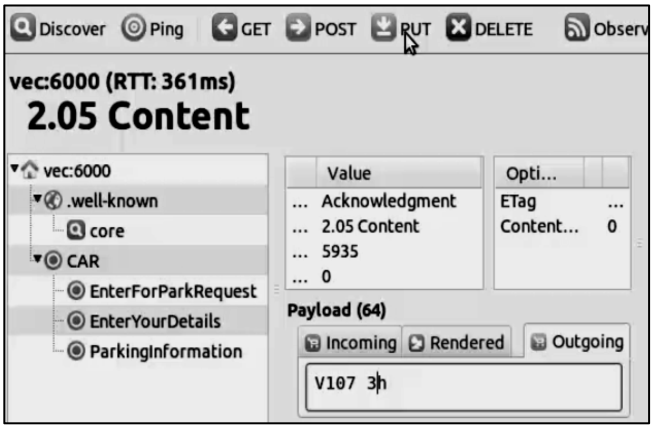

Fig.9(a). Input is given via Outgoing as V107 3h

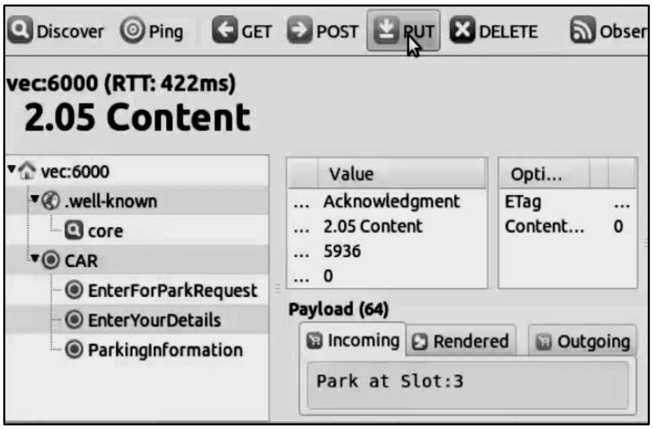

Fig.9(b). Output – Free slot is displayed via Incoming tab

The Fig.9(a) shows that the input is given via the resources in the outgoing tab as V107 3h whereas V shows the Vehicle number and h reveals the parking time in hours. Fig.9(b) shows the output as per the proposed algorithm. As the parking time is below 5 hours moreover slots 1 and 2 are already filled hence the output is shown as Park at Slot 3. The intention is to check out for the parking time based on the acuters wish.

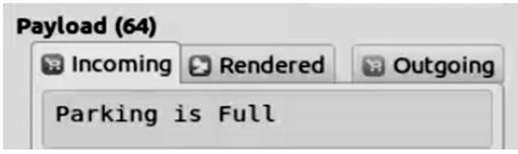

Fig.10. Status of the Parking Lot

After the entry of 10 vehicles, the browser reminds the user as Parking is full because the parking lot is capable of ten slots is shown in Fig.10. Various scenarios have been depicted for the car parking application as shown in Table.3.

Table.3. Outputs based on various scenarios

SI. No.	Conditions	Outgoing	Incoming
1	>=6 hours	V105 7H	Park at 6
2	<=5 hours	V108 3H	Park at 4
3	>=10 vehicles	V111 2H	Parking Full

Performance Analysis is done to view the efficiency of the proposed work. CoAP performance can be analyzed by WireShark network protocol Analyzer [20], MSPSim [21].

Table.4. Performance Analysis

SI.No.	Method/Tools	Parameters	Values
1	M/M/1:N/SIRO(TIME)	Parking Time	3 minutes
2	Wireshark	PDR	100 %
3	MSPSim	CPU Usage	1.62 %

The Table.4 reveals that the performance analysis of the proposed paper. Parking time is decreased to 3 minutes from <= 10 minutes,Packet Delivery Ratio(PDR) stays constant and CPU usage from >= 3% to that of 1.62 % by the proposed algorithm.

6. CONCLUSION AND FUTURE WORK

Thus a Graphical User Interface is created via CoAP address than *http* address as a gateway between the driver of the car and the parking lot manager. CoAP has the following advantages: asynchronous in nature, designed for 6LoWPAN (IPv6 over Low Power Wireless Personal Area Networks), multicasting, building resource discovery, and security. CoAP is analyzed for its efficiency over HTTP. The searching time of the driver gets decreased comparatively. Real time implementation and deploying this application in cloud with high security factor will be the future work.

REFERENCES

[1] Kaivan Karimi and Gary Atkinson, “What the Internet of Things (IoT) Needs to Become a Reality”, *White Paper, FreeScale and ARM*, 2013.

[2] M. Cops, “Vehicle Infrastructure Integration (VII) – Bringing Vehicle Adhoc Network Technology on the road”, *Third ACM International Workshop on Vehicular Ad Hoc Networks*, 2006.

[3] Jae Kyu Suhr and Ho Gi Jung, “Fully-automatic recognition of various parking slot markings in Around View Monitor (AVM) image sequences”, *15th International IEEE Conference on Intelligent Transportation Systems*, pp. 1294-1299, 2012.

[4] Ho Gi Jung, Dong Suk Kim, Pal Joo Yoon and Jaihie Kim, “Parking slot markings recognition for automatic parking assist system”, *IEEE Intelligent Vehicles Symposium*, pp. 106-113, 2006.

[5] Luis Ostiz, Urdiain, Carlos Pita Romero, Jeroen Doggen, Tim Dams and Patrick Van Houtven, “Wireless Sensor Network Protocol for Smart Parking Application Experimental Study on the Arduino Platform”, *Second International Conference on Ambient Computing, Applications, Services and Technologies*, pp. 45-48, 2012.

[6] Jonathan P. Benson, Tony O'Donovan, Padraig O'Sullivan, Utz Roedig, Cormac Sreenan, John Barton, Aoife Murphy and Brendan O'Flynn, “Car-park management using wireless sensor networks”, *Proceedings of 31st IEEE Conference on Local Computer Networks*, pp. 588-595, 2006.

[7] Libelium, “Smart Parking: Technical Guide”, Available at: www.libelium.com, 2013.

[8] R. Aarthi and A. Pravin Renold, “Car parking Applications based on sensor networks: A Survey”, *Proceedings of the TEQIP II Sponsered National Conference on Wireless Networks Computing and Communication*, pp. 30-36, 2014.

[9] Paventhan, Sai Krishna Allu, V. Gayathri, Sameer Barve and N. Mohan Ram, “Leveraging CoAP towards monitoring agriculture sensors network”, *ERNET India R&D Centre*, 2012.

[10] R.P. Vinob Chander, Susan Elias, Subramanian Shivashankar and P. Manoj, "A REST based design for Web of Things in smart environments", *2nd IEEE International Conference on Parallel Distributed and Grid Computing*, pp. 337-342, 2012.

[11] Tomislav Dimcic, Srdan Krco and Nenad Gligoric, "CoAP (Constrained Application Protocol) implementation in M2M Environmental Monitoring System", *E-society Journal*, Vol. 21, 2012.

[12] Berta Carballido Villaverde, Dirk Pesch, Rodolfo De Paz Alberola, Szymon Fedor and Menouer Boubekeur, "Constrained application protocol for low power embedded networks: A survey", *Sixth IEEE International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, pp. 702-707, 2012.

[13] Matthias Kovatsch, "CoAP for the web of things: from tiny resource-constrained devices to the web browser", *Proceedings of ACM conference on Pervasive and ubiquitous computing adjunct publication*, pp. 1495-1504, 2013.

[14] Constrained Application Protocol (CoAP), "draft-ietf-core-coap-18", 2013.

[15] Mozilla Firefox add-ons, "Copper", Available at: http://people.inf.ethz.ch /mkovatsc/copper.php, 2013.

[16] W.J. Anderson, "*Continuous-time Markov Chains: An Applications-Oriented Approach*", Springer Series in Statistics, 1991.

[17] W. Stewart, "*Introduction to the Numerical Solution of Markov Chains*", Princeton University Press, 1994.

[18] Caliskan, Murat, Andreas Barthels, Bjorn Scheuermann and Martin Mauve, "Predicting parking lot occupancy in vehicular ad hoc networks", *65th IEEE Conference of Vehicular Technology*, pp. 277-281, 2007.

[19] F. Osterlind, Adam Dunkels, Joakim Eriksson, Niclas Finne and Thiemo Voigt, "Demo abstract: Cross-level simulation in cooja", *Proceedings of the First IEEE International Workshop on Practical Issues in Building Sensor Network Applications*, 2006.

[20] Ulf Lamping, Richard Sharpe and Ed Warnicke, "WireShark user's guide", 2004.

[21] Joakim Eriksson, Adam Dunkels, Niclas Finne, Fredrik Osterlind and Thiemo Voigt, "Mspsim–an extensible simulator for msp430-equipped sensor boards", *Proceedings of the 4th European Conference on Wireless Sensor Networks*, 2007.