CoAP Based Wireless Sensor Network Design for Effluent Treatment of Water in Textile Industries

Adarsh.B.U
Dept. of Telecommunication Engg
M.S Ramaiah Institute of Technology
Bangalore, Karnataka, India
adarsh.b.u91@gmail.com

Dr.K.Natarajan
Dept. of Telecommunication Engg
M.S Ramaiah Institute of Technology
Bangalore, Karnataka, India
drknatraj@msrit.edu

Shobha.K.R
Dept. of Telecommunication Engg
M.S Ramaiah Institute of Technology
Bangalore, Karnataka, India
shobha_shankar@msrit.edu

Abstract—**South India has many Industries where large scale textile activity is being carried out. Because of the impurities let out by the textile industries the groundwater quality around the industries is getting polluted to such a level that it is unfit for domestic, industrial and agricultural activities. Further, pollution has adversely impacted human health increasing water-borne diseases such as diarrhea, typhoid, malaria, jaundice and skin allergies. Pollution due to textile industries also has adverse impact on the agricultural yields, cattle, animals, birds, fish and other aquatic life. The quality monitoring of effluent water can be performed using manual chemical analysis, which has many disadvantages like it is time consuming and expensive, it needs to be measured on-site to ensure accuracy etc. A solution has been provided in this paper to tackle this problem. Objective of this work is to design and implement a CoAP based smart and intelligent system that can collect the data from water that is let out of the industry and transmit the data wirelessly to a main system or the user, using internet. Contiki OS is used for the simulation environment and TelosB motes are used for hardware implementation.**

Keywords: *IEEE 8 02.11, IOT Protocols, quality monitoring, 6LoWPAN, CoAP.*

I. Introduction

In many developing countries like India, rapid growth in terms of industrialization, urbanization, population amplifies the consumption of water. Increase in usage of water-intensive products further enhances overall usage of water. With rising consumption, deteriorating water quality and inadequate governance, India is likely to face severe water shortage. The textile industries have great economic significance by virtue of its contribution to overall industrial output and employment generation. This sector has wide spectrum of industries ranging from small scale units that use traditional manufacturing process to large integrated mills using modern machineries and equipment. The textile units use a number of dyes, chemicals and other materials to impart desired quality to the fabrics. These units generates a substantial quantity of effluents, the quality of which in most of the cases is unsuitable for further use and can cause environmental problems, if disposed off without proper treatment. To provide a solution this problem, a remote real time CoAP based water quality monitoring system using wireless sensor network has been designed and implemented in this project.

The Internet of Things (IoTs) is an automated communication model used to define the relationship between communicating constrained devices such as wireless sensors and their applications. The IoTs adaptation has become more noticeable and realistic and was significantly facilitated by the deployment of IPv6, as the large address space provided in IPv6 enables even more machines to be accessible over the Internet and thus, communicate with each other. According to World Wide Web Consortium (W3C) [1], there are two approaches to realize and manipulate web services, the Representation State Transfer (REST) and the arbitrary Web Services.

REST is an approach that makes information available on a server and accessible to clients through Uniform Resource Identifiers (URI). Clients communicate with the server synchronously in a request /response manner using methods of a transfer protocol such as the Hypertext Transfer Protocol (HTTP) [2]. HTTP is a predominant protocol for the web services and online applications. However, incorporating the IoTs into traditional Internet is not a straightforward operation; this is due to the differences between the implementation model of the IoTs-based applications and the current applications in the Internet.

For instance, most Internet applications that use HTTP protocol depend on the Pull Model [3] for data exchange whereas inactive nodes in the IoTs will be put to sleep mode to save the battery life and extend the life cycle of the node. These nodes only wake up to perform specific tasks when required. Therefore, it's obvious that the Pull model will not be suitable for constrained networks. In order to extend IoTs to the traditional Internet, a fresh approach is unavoidable to make integration process more practical and viable. The need for a new integration approach stemmed from the fact that the current technologies/protocols utilized by current web services such as HTTP, Transmission Control Protocol (TCP), Simple Object Access Protocol (SOAP) [4], and Extensible Markup Language (XML) [5] are not appropriate for such facilitation. This has been realized by different research groups, which have proposed mechanisms to facilitate the communications between constrained devices within the IoTs model. Examples of these new technologies are the IPv6 over Low Power

Wireless Personal Area Networks (6LowPAN) [6] as a network level enabler, the Routing over Low-power and Lossy networks (ROLL) [7] to provide a routing mechanism optimized for constrained networks, and the Constrained Application Protocol (CoAP) as an application layer enabler [8]. This paper provides an approach and prototype implementation for effluent treatment of water in textile industries, which is an IoT based application.

The remainder of this paper is organized as follows: Section II describes the related work. Section III gives the description of the system and the protocols required for the implementation. Section IV describes the hardware and software required to set up the test bed. Details of the prototype implementation are given in section V. Finally, section VI presents the conclusion and future work.

II. RELATED WORKS

Wireless Sensor Network (WSN) platform developed by "Smart Coast" Multi Sensor System for water quality monitoring [10] allows for the integration of the different sensors like water-temperature, phosphate, dissolved oxygen, conductivity, pH, turbidity and water level. The system uses Zigbee communication to meet the low power requirements of the deployment scenario. The Smart Coast project also developed a portable, field deployable sensor for long-term monitoring of phosphate levels in natural water which was set with a limit of detection of 0.3 mg/L. For resolving the problem of the manual analytical method adopted in water quality detection with lack of real-time data, a novel type of remote water quality measuring and monitoring System based on WSN [9] is implemented, where the sensor nodes in the system enter the sleep mode when it does not collect the data so as to reduce the power consumption.

Another approach [11] explains the development and testing of a multi-sensor heterogeneous real-time water monitoring system which was deployed in the River Lee Co. Cork, Ireland to monitor water quality parameters such as pH, temperature, conductivity, turbidity and dissolved oxygen. Sensor interface infrastructure (incorporating Programmable System on Chip - PSOC technology) and data telemetry systems compatible with the Zigbee data transmission system were developed which is capable of transmitting the data to the Smart Coast server, which processed the data for transmission to the web. The PSOC system accommodates the output magnitude from the sensors and processes the data in order to make it generic for the communication and processing unit of the system.

A water quality measurement system based on thick-film technology [12] is used for sensing pH, temperature, oxidation reduction potential (ORP), conductivity and turbidity. Other sensors are being developed for, organic contamination, dissolved oxygen and diversions. Here, the entire system is coated with a transparent resin, with the exception of the sensor area which must be in direct contact with the water, for serving the purpose of sensing.

III. DESCRIPTION OF THE SYSTEM

The proposed system for effluent treatment of water is shown in figure 1. Routing protocols for sensor networks are often designed with explicit assumptions, serving to simplify design and reduce the necessary energy, processing and communications requirements. The basic protocols used for the implementation of the system are discussed in this section.

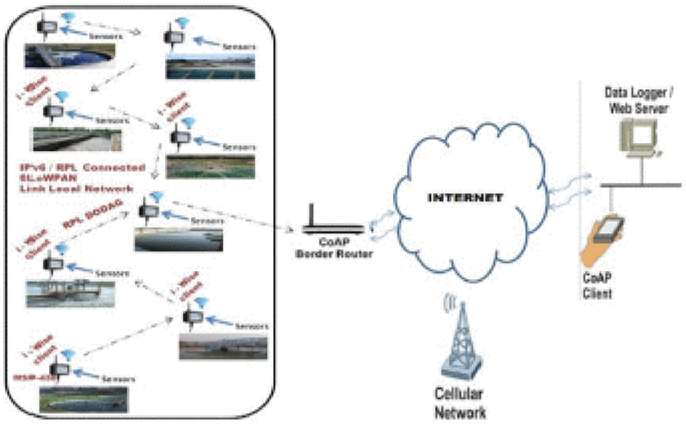


Figure 1: Proposed System for effluent treatment of water

A. *Routing Protocol for Low power and Lossy Networks (RPL)*

RPL is on the IETF standards track for routing in low-power and lossy networks [13, 14]. The protocol is tree-oriented in the sense that one or more root nodes in a network may generate a topology that trickles downward to leaf nodes. Much flexibility is given in RPL as to the constraints and metrics that can be used when building a topology.

Topology Formation:
A network that manages its routing topologies using RPL may run one or more RPL instances. Each instance defines a topology that is built using a unique metric or constraint within the network. In each RPL instance, multiple Directed Acyclic Graphs (DAGs) may exist, each having a different DAG root. A node may join multiple RPL instances, but must only belong to one DAG within each instance. The operation of multiple RPL instances is shown in Figure 2. When forming the topology, each sink constructs a packet called a DAG Information Object (DIO), and sends it to all children. Any child that decides to join the DAG may pass the DIO further to its own children. The DIO contains a rank that is increased monotonically when the child joins the DAG. The rank prevents routing loops, and helps the nodes to distinguish between parents and siblings. Nodes may store a set of candidate parents and siblings that can be used if the preferred parent is unable to route traffic for the node.

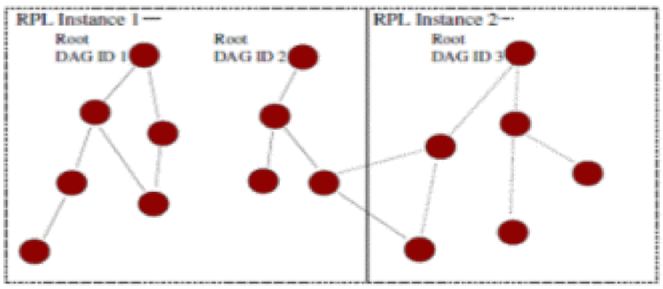


Figure 2: A network with three DAGs in two RPL instances.

Maintenance:
DAGs may need to change if the network restructures because of mobility or link quality variance. RPL ensures that DAGs are adjusted occasionally by having the root send out a new DAG iteration. A Trickle timer regulates when nodes should forward such information to their children, which efficiently suppresses many redundant updates in dense networks [15].

Point-to-Point Traffic:
To let arbitrary nodes communicate with each other, RPL includes a Destination Advertisement Object (DAO) in which children can advertise their own address prefixes to their neighbourhood using multicasts, or back to the DAG root using unicasts after joining a DAG.

B. IPv6 over Low power WPAN (6LoWPAN)

The IETF 6LoWPAN working group has defined specifications [16, 17] to efficiently transport IPv6 datagrams over IEEE 802.15.4 links. The Internet Protocol (IP) is predominantly used over Ethernet links that offer increasingly high throughput. The transmission of IPv6 packet over LoWPAN links are faced with several challenges due to the resource constraints

IETF RFC 6282 [17] defines an adaptation layer considering that the minimum IPv6 MTU (1280 bytes) is much larger than the largest 802.15.4 frame size (127 bytes). The major functions of the 6LoWPAN adaptation layer that works between IPv6 and 802.15.4 MAC layer are listed below.

Fragmentation:
Considering IPv6 MTU size (1280 bytes) and 802.15.4 frame size (127 bytes), the fragmentation and reassembly is essential at the 6LoWPAN adaption layer. The fragmentation header includes IPv6 datagram size, datagram offset and a datagram tag to help in reassembly.

Header compression:
The header compression is important to increase the effective pay load of the upper layers. For example, 40 bytes of IPv6 header is compressed to 3 bytes as the source-, destination address and datagram size can be inferred from layer 2 header.

Layer 2 forwarding:
In mesh routing, the source and the final destination need not be directly connected. The mesh header carries the source and final destination link layer addresses and hop count. The intermediate node forwards the packet to the next hop after deducting the hop count. The Reduced Function Device (RFD) relies on the Full Function Device (FFD) for datagram forwarding.

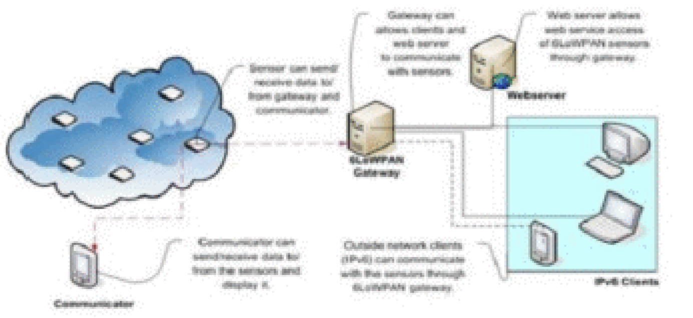

Sensor can send/ receive data to/ from gateway and communicator.
Gateway can allows clients and web server to communicate with sensors.
Web server allows web service access of 6LoWPAN sensors through gateway.
Webserver
6LoWPAN Gateway
Communicator can send/receive data to/ from the sensors and display it.
Outside network clients (IPv6) can communicate with the sensors through 6LoWPAN gateway.
IPv6 Clients
Communicator

Figure 3: 6LoWPAN typical architecture

Standard 6LoWPAN architecture consists of several entities. Figure 3 illustrates the typical 6LoWPAN architecture in which 6LoWPAN gateway is a primary source for outside the network IPv6 clients to communicate with 6LoWPAN sensor nodes. Whereas it also shows web server, which retrieves sensor data from the 6LoWPAN gateway and publishes on the Internet.

C. Constrained Application Protocol (CoAP)

CoAP is a Web-oriented protocol, i.e., it adopts features from HTTP, the Web protocol. The central ones are the resource abstraction, RESTful interaction [18], and extensible header options. These features allowed the Web to evolve from a simple document retrieval mechanism ('World- Wide Web') to a thriving application platform ('Web 2.0'). Being a successful, long-lived IETF standard, HTTP allows us to combine different resources or services with very little scripting effort in so called 'mashups.' This interoperability is the key argument for the 'Web of Things' initiative [19] to push HTTP down to the device level. HTTP over TCP, however, has a one-to-one communication model, no native push notifications, and is rather heavy-weight for constrained devices.

To overcome this downside, CoAP enables size-optimization and reliable datagram communication. On one hand, it offers URIs (e.g., coap://vs0.inf.ethz.ch/), the RESTful methods GET, POST, PUT, and DELETE, and extensions through header options that can be defined independently. On the other hand, CoAP uses UDP, which is lighter than TCP, but moreover allows for efficient IP multicast. Group communication is a significant requirement for the Internet of Things. To make up for the unreliability of UDP, CoAP defines transactions with retransmissions. Native push notifications for eventing are supported with the publish/subscribe pattern to observe resource changes [20]. Finally, a resource discovery mechanism is provided, which also provides for resource descriptions.

IV. TEST BED SETUP

A. WIRELESS SENSOR NETWORK HARDWARE

Hardware design of the system consists of a sensor unit and a wireless sensor node. Sensor unit consists of the sensors that sense the critical parameters like turbidity (determines cloudiness or haziness of a fluid caused by individual particles), pH (measure of the acidity or basicity of an aqueous solution), conductivity (ability to conduct or transmit heat, electricity, or sound), temperature (Temperature is a measure of the average energy of water molecule) and colour. All sensors are calibrated to provide a DC output voltage which ranges between 0 to 2V.

TelosB mote was used as wireless sensor node. TelosB is an ultra low power wireless module for use in sensor networks, monitoring applications, and rapid application prototyping. It has integrated onboard antenna with 50m range indoors / 125m range outdoors, 16-pin expansion support and optional SMA antenna connector. TelosB leverages industry standards like USB and IEEE 802.15.4 to interoperate seamlessly with other devices. TelosB enables a wide range of mesh network applications. The low power operation of the TelosB module is due to the ultra low power Texas Instruments MSP430 F1611 microcontroller featuring 10kB of RAM, 48kB of flash, and 128B of information Storage. TelosB features the 250kbps 2.4GHz IEEE 802.15.4 Chipcon CC2420 radio for wireless communications. The sensors mentioned above are integrated with the mote. TelosB may be powered by two AA batteries.

B. SIMULATION ENVIRONMENT

Software simulation environment is Contiki 2.7, an operating system developed for constrained environments. A Contiki system is partitioned into two parts: the core and the loaded programs. Typically, the core consists of the Contiki kernel, the program loader, the most commonly used parts of the language run-time and support libraries, and a communication stack with device drivers for the communication hardware. The core is compiled into a single binary image that is stored in the devices prior to deployment. The core is generally not modified after deployment, even though it should be noted that it is possible to use a special boot loader to overwrite or patch the core. Programs are loaded into the system by the program loader. Programs to be loaded into the system are first stored in EEP-ROM before they are programmed into the code memory. CoAP implementation that is bundled with Contiki 2.7 is used in our development. The REST engine defined in Contiki 2.7 includes framework for developing both CoAP server and CoAP client applications. Firefox browser plugin, Copper (Cu), a CoAP user agent implementation for monitoring resources using Web browser, is included in this framework. CoAP resources are defined using the RESOURCE macro.

V. PROTOTYPE IMPLEMENTATION

The prototype implementation of effluent water monitoring system is shown in the figure 4a and figure 4b. CoAP resources in copper are defined using the RESOURCE macro. For example, the following macro defines a temperature sensor resource.

```
RESOURCE(temperature, METHOD_GET, "temperature");
void temperature_handler(REQUEST* request, RESPONSE*
response)
```

The parameters defined in the above example are, name of the resource, request methods supported, URI path string, title and the resource type. Similarly, the conductivity resource of water sensor can be defined using the following macro.

```
RESOURCE(conductivity, METHOD_GET, "conductivity");
void conductivity_handler(REQUEST* request, RESPONSE*
response)
```

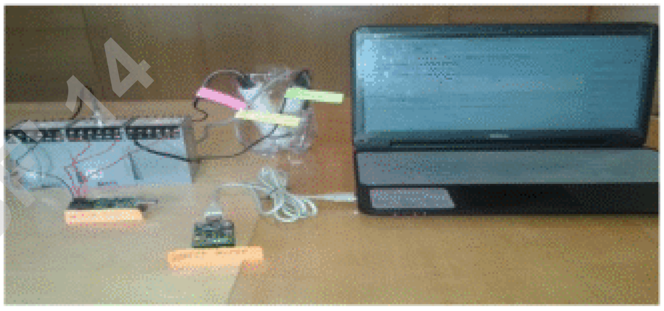


Figure 4a: Prototype Implementation

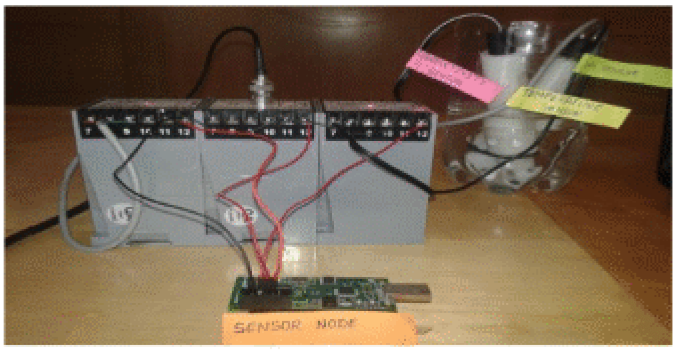
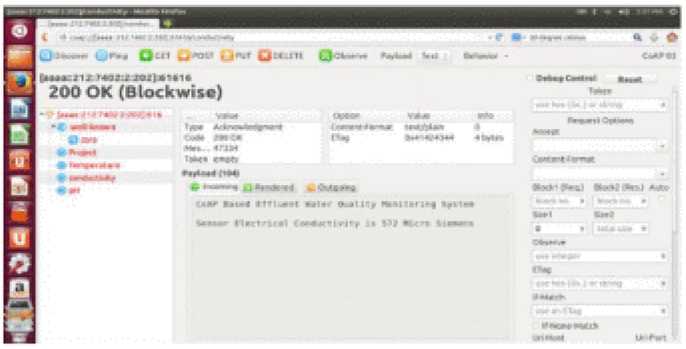


Figure 4a: Sensors and Mote

All the resources are statically defined and the associated functions are registered when the REST engine in CoAP server is started. Each resource has to implement a handler function with the name [resource_name]_handler. For example, when the CoAP client sends a GET request to the Copper server for the CoAP URI /water-sensor/conductivity,

conductivity_handler function will be called by the Copper REST engine. The handler implements actions for sending an excitation signal to water sensor that results in reading of the conductivity value. The CoAP response message is formatted according to the client requested format, which can be plain text, xml or JavaScript Object Notation (JSON).

Figure 5 shows CoAP client plugin in Mozilla web browser displaying the 2.05 Content response received for a GET request. The browser displays individual icons for each request methods like GET, POST, PUT and DELETE. The panel on the left showing the expanded list of resources for the DISCOVER request.



[aaaa:212:7402:2:202]:61616
200 OK (Blockwise)

Type Acknowledgment
Code 2.05 OK
Token empty
Option Content-Format text/plain
Payload (104)

```
CoAP Based Effluent Water Quality Monitoring System

Sensor Electrical Conductivity is 572 Micro Siemens
```

Figure 5: CoAP client plugin

VI. CONCLUSION AND FUTURE WORK

In our project, we have demonstrated how to automate the process of fetching water quality parameters using wireless sensor network. This prototype overcomes the disadvantages of manual chemical analysis and also provides a solution for the problems caused due to effluent water let out by the textile industries. Further extension of this project is, its real time implementation in effluent industries. The prototype can also be implemented by considering other effluent water monitoring parameters and suitable sensors.

ACKNOWLEDGMENT

This project was funded by KSCST and supported by Dept of TCE-MSRIT, Dept of ECE-IISc and DeitY-ERNET, Bangalore, India.

REFERENCES

[1] D.Booth, H.Haas, F.McCabe, E.Newcomer, M.Champion, C.Ferris. "Web Services Architecture". W3C Working Group Note 11 February 2004. Available: http://www.w3.org/TR/ws-arch

[2] R. Fielding and UC Irvine, Hypertext Transfer Protocol -- HTTP/1.1, RFC 2616, June 1999.

[3] Z. Shelby, "Embedded web services" IEEE Wireless Communications, vol. 17, no. 6, pp. 52–57, 2010.

[4] M. Gudgin, M. Hadley, N. Mendelsohn, Jean-Jacques Moreau, H.Nielsen, A. Karmarkar, Y. Lafon. "Simple Object Access Protocol (SOAP)". Version 1.2 Part 1: Messaging Framework (2nd Ed.). Available: http://www.w3.org/TR/soap12-part1/

[5] Extensible Markup Language (XML) Available: http://www.w3.org/XML/

[6] N. Kushalnagar, G. Montenegro, C. Schumacher. "IPv6 over Low Power Wireless Personal Area Networks" [R FC 4944]. Aug 2007 Available: http://tools.ietf.org/html/rfc4944.

[7] "Routing Over Low power and Lossy networks" ROLL Working Group. Available at: http://datatracker.ietf.org/wg/roll/

[8] Z. Shelby, B. Frank, and D. Sturek. "Constrained Application Protocol (CoAP)". Internet-Draft (work in progress draft-ietf-core-coap-018), Sensinode, SkyFoundry, Pacific Gas and Electric, June 2013.

[9] Zhu Wang, Qi Wang, Xiaoqiang Hao, "The Design of the Remote Water Quality Monitoring System based on WSN" Proceedings of the IEEE 2009.

[10] B O'Flynn, Rafael Martínez-Català, S. Harte, C. O'Mathuna, John Cleary, C. Slater, F. Regan, D. Diamond, Heather Murphy, "SmartCoast-A Wireless Sensor Network for Water Quality Monitoring" 32nd IEEE Conference on Local Computer Networks, pp. 815 - 816, 2007.

[11] Fiona Regan, Antoin Lawlor, Brendan O Flynn, J. Torres2, R Martinez-Catala2, C. O'Mathuna, John Wallace, "A demonstration of wireless sensing for long term monitoring of water quality" 4th IEEE International Workshop on Practical Issues In Building Sensor Network Applications, Zurich, pp. 819 - 825, October 20-23, 2009.

[12] Eduardo Garcia Breijo, Luis Gil Sanchez, Javier Ibañez Civera, Alvaro Tormos Ferrando, Gema Prats Boluda, "Thick-Film Multisensor for Determining Water Quality Parameters" Proceedings of the IEEE, 2002.

[13] J. Vasseur and A. Dunkels. Interconnecting Smart Objects with IP: The Next Internet. Morgan Kaufmann, 2010.

[14] T. Winter (Ed.), P. Thubert (Ed.), and the ROLL Team. RPL: IPv6 Routing Protocol for Low power and Lossy Networks

[15] P. Levis, N. Patel, D. Culler, and S. Shenker. Trickle: A self-regulating algorithm for code propagation and maintenance in wireless sensor networks. In Proceedings of the USENIX Symposium on Networked Systems Design & Implementation (NSDI), Mar. 2004.

[16] G. Montenegro, N. Kushalnagar, J. Hui and D. Culler, Transmission of IPv6 Packets over IEEE 802.15.4 Networks. IETF RFC 4944, September 2007.

[17] J. Hui and P. Thubert, Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks. IETF RFC 6282, September 2011.

[18] R. T. Fielding. Architectural Styles and the Design of Network-based Software Architectures. Phd thesis, UC Irvine, 2000.

[19] D. Guinard, V. Trifa, and E. Wilde. A Resource Oriented Architecture for the Web of Things. In Proc. IoT, Tokyo, Japan, 2010

[20] K. Hartke and Z. Shelby. Observing Resources in CoAP. draft-ietfcore-observe-02, 2011.