Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 94 (2016) 137 – 143

Procedia Computer Science

The 13th International Conference on Mobile Systems and Pervasive Computing (MobiSPC 2016)

Management and Internet of Things

Mayra Samaniego, Ralph Deters*

mayra.samaniego@usask.ca, deters@cs.usask.ca

University of Saskatchewan, Dep. Computer Science, Saskatoon, Canada

---

Abstract

The study and development of Internet of Things (IoT) applications, web and mobile, is on the increase. Applications, working with data obtained from different areas such as transportation, smart homes, health care, public services, industry and many others. Previous studies have focused on managing the obtained data. However, managing the heterogeneous resources that get that data is an area that demands more attention. This work addresses the management of resources in the Internet of Things. This is achieved by proposing a virtual-resource edge layer, which enables access and configuration to constrained physical resources. The architecture presented focuses on the use of virtual resources as a management concept and identifies different approaches in the performance evaluation on edge computing devices. Using the IoT protocol CoAP, virtual resources are exposed in the edge network. An evaluation of a Go CoAP virtual resource is presented.

Peer-review under responsibility of the Conference Program Chairs

Keywords: Internet of Things; IoT Management; Virtual Resources; CoAP

---

1. Introduction

In the Internet of Things[1,2], the communication and resource management have been performed based on the technology in which the constrained network is implemented [6,7,8]. Each vendor has made sure of developing its own encapsulated-compatible and propietary communication standards. Applications have been supported by those propietary standars to get reliable information in an efficient and secure manner from the constrained environment to the cloud [3,4,5]. This has influenced users to adopt a certain technology in entire implementations to facilitate the duties of management (ITU-T M.3400). There has been some studies on facing IoT physical devices, but their main goal has been to facilitate the job of developers [10,11]. Nowadays, there is not a solution to allow access to diversified IoT network resources to get their information. This causes limitations in fulfilling the vision of the Internet of Things[9]. This work extends that vision not only having interconnected heterogenous "things", but also getting information in real time about those "things" that are working in the environment. The architecture put forward is focused on abstracting the complexity of access and configuration of heterogeneous constrained networks resources to the concept of virtual

---

* Corresponding author. Tel.: +1-306-966-2072 ; fax: +1-306-966-4884.
E-mail address: deters@cs.usask.ca

Peer-review under responsibility of the Conference Program Chairs
doi:10.1016/j.procs.2016.08.022

resources, offering a simplified management option. This work proposes the creation of virtual resources running in edge computing devices. Also, this study adopts the Constrained Application Protocol (CoAP) to communicate the virtual resources and the Go programming language to build them. This representation allows constrained components to be easily accessed from third parties. The direct communication with physical resources is broken up introducing virtual resource layers.

The remainder of the paper is structured as follows. Section 2 presents the definition of our IoT resources. Section 3 explains the experimental setup. Section 4 presents the performance evaluation of our prototype and the paper concludes with a summary in section 5.

2. IoT Virtual Resource Architectural Design

We define an IoT virtual resource as an abstraction of other resources, virtual or not. The architectural design of the IoT virtual resources is graphically illustrated in Fig. 1. Using the REST features of CoAP (RFC7252), a virtual resource exposes its own status and methods that interact with the states and can resolve interactions between its compound resources. The entire architecture comprises of three major layers namely: the view abstraction layer (VAL), the hardware abstraction layer (HAL) and the physical layer. The physical layer represents the sensors working in the environment. The two layers, hardware abstraction and view abstraction, are edge-hosted application middlewares, which are aimed at encapsulating the complexicity of access and configuration of physical resources. The HAL represents the lowest-level association between a physical resource and a virtual resource, resulting in a one-to-one relation, see Fig. 2. The virtual resources placed in this layer make use of CoAP verbs to receive orders to perform operations over their faced physical resources. The VAL acts as a bridge between final users and sensors. It represents an association between one or more HAL virtual resources and one or more VAL virtual resources, resulting in a many-to-many relation, see Fig. 3. The purpose of the virtual resources placed in this layer is to provide dynamic views to different kind of users with different access privileges. Also, this layer works as an edge-processing center in which the row-data is evaluated without going to the cloud. In this layer, virtual resources are divided into two groups, state-less and state-full virtual resources. State-less virtual resources, see Fig. 4, make use of the CoAP REST pattern to pull the states each time a request is received. State-full virtual resources, see Fig. 5, maintain the states of their compound resources in a database and make use of the CoAP observe pattern to receive updates of the resources they are observing.

This architecture is intended to cross the barrier of having only one level of communication, physical, when working with sensors.

3. Experimental Setup

This experiment was run over a VAL resource. The prototype involves a constrained network infrastructure, virtual resources and emulated clients. Once the VAL resource receives a CoAP request from the CoAP client, it gets the state of its compound resources and sends the processed information in a CoAP message. A Raspberry Pi with Go language and CoAP protocol was used as the edge computing layer, to run the virtual resource, see Fig. 6. We built the virtual resource and clients using the channels and routines features of Go, see Fig. 7. We set up the proposed architecture with devices with the following specifications: Raspberry PI 2 Model B with. processor ARMv7, CPU: ARM Cortex-A7 quad core - 900 MHz, RAM: 1GB, Operating system: Raspbian OS. IMac with processor: Intel Core i7 - 3.5 GHz. RAM: 16 GB, 64 bits operating system. The clients connected to the virtual resource over a dedicated WiFi connection.

4. Evaluation

To evaluate the usefulness of our virtual resource we measured its throughput and response times. We evaluated the behavior of our virtual resource testing two processes, the Core Link Format (RFC6690) discovery of services through a well-know interface and the current state retrieving. These evaluations were performed from the client perspective. Also, we evaluated the database connection time in the VAL state-full resource. The services that our virtual resource

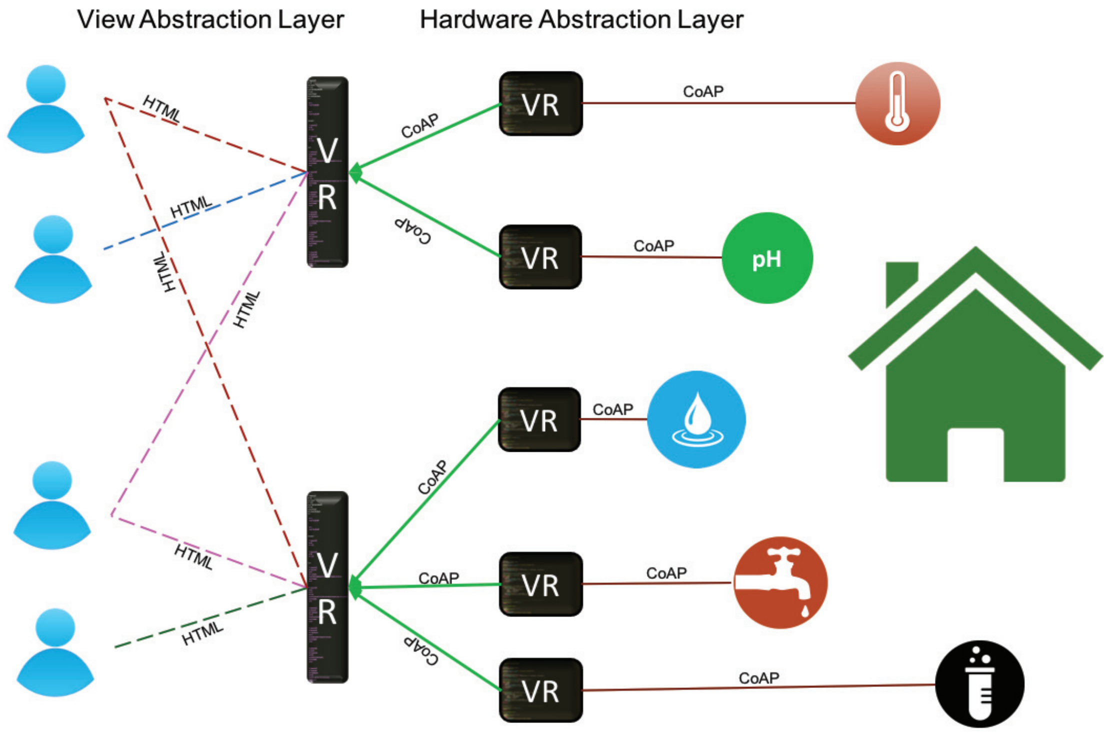

Fig. 1. The architectural design of the IoT virtual resources. Example of the IoT ecosystem in a green house incorporating the virtual resources.

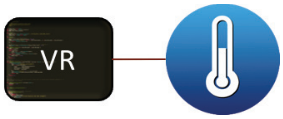

Fig. 2. Virtual Resource One-To-One Relation.

exposed were hosted as CoAP REST services. The scenario consisted of a Go CoAP Virtual Resource running in a Raspberry Pi and one thousand of Go CoAP virtual clients, running in IMac computer. This experiment introduced delay times in the client requests from 0 to 100 milliseconds, with intervals of 50 milliseconds, emulating different levels of concurrency.

The results for all one thousand requests in each delay scenario are explained as follows. The average of the discovery of service round trip time (RTT) for the three delay cases, 0, 50 and 100 ms, are 2.24 ms, 2.40 ms and 2.33 ms respectively, see Fig. 8. In this test the workload of the virtual resource and its response times remain stable for the three delay cases. The average RTT time to get the current state of the virtual resource for the three delay cases are 12.01 ms, 14.47 ms and 12.98 ms. In this process, the average response time increased as this is a state-full virtual resource that has to go to the database to get the current state of its compound resources, see Fig. 9. The average time that the virtual resource spends going to the database to search the state of an specific compound resource in the three delay times are 7.53ms, 8.29 ms and 8.07 ms, see Fig. 10. In all three evaluations there were peaks that can be attributed to the network traffic and backup processes, which represents common anomalies. The database ran in a separated machine.

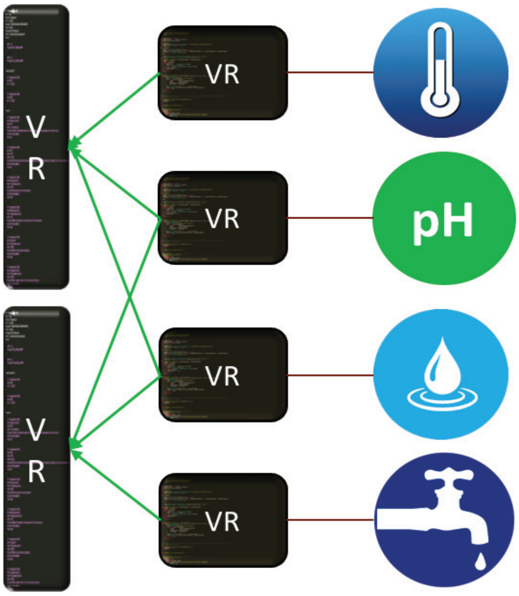

Fig. 3. Virtual Resource Many-To-Many Relation.

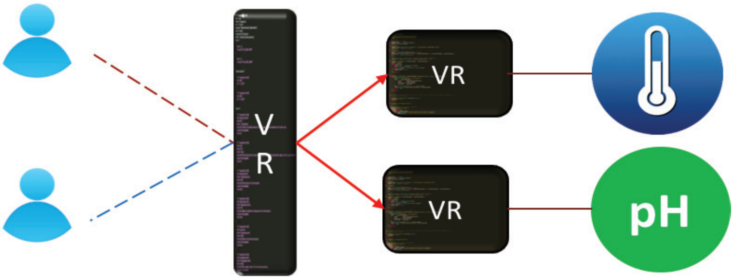

Fig. 4. Representation of a State-Less Virtual Resource.

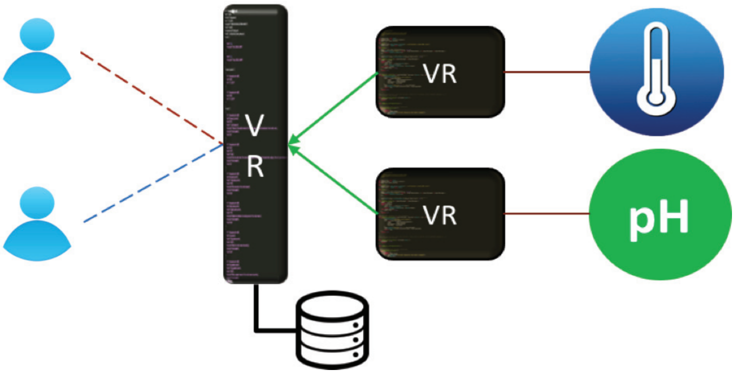

Fig. 5. Representation of a State-Full Virtual Resource.

5. Conclusions

This work abstracted the complexity of IoT resource access and configuratioin to an edge computing layer that is managed via CoAP protocol. Our virtual resource performed in an expected manner, responding to all requests no

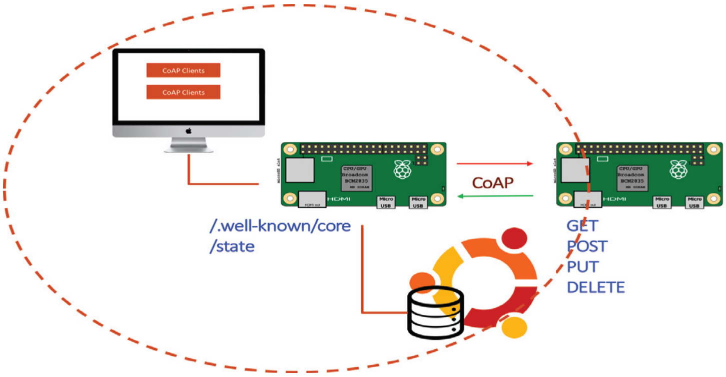

Fig. 6. Experimental Set Up

```go
func handle_request(m *coap.Message) (payload []byte, err error) {
    code := m.Code
    switch code {
    case coap.GET:
        payload, err = handle_GET("temperature")
    case coap.DELETE:
        chan_stop <- true
    }
    return
}
```

```go
func listening_to_exit() {
    for {
        select {
        case <-chan_stop:
            log.Println("I am leaving...  god bye")
            os.Exit(0)
        }
    }

}
```

Fig. 7. Golang Code Example

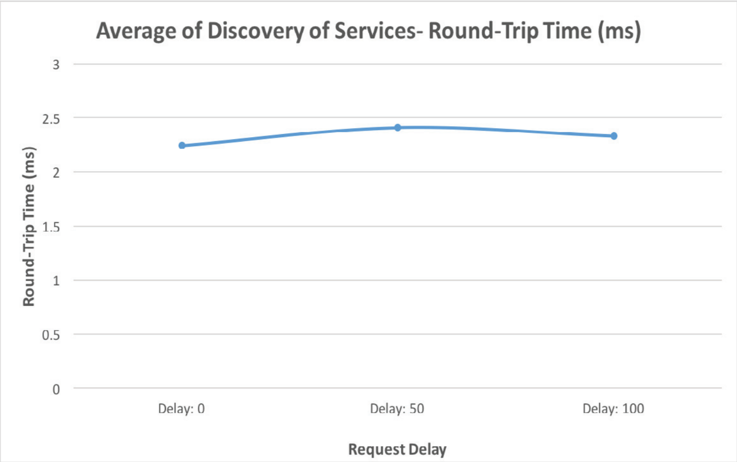

Fig. 8. One thousand of virtual clients. Virtual Resource Discovery-of-Services Process Round Trip Time.

matter the concurrency level they faced. The virtual resource layer could interact between each compound components as easy as consuming a RESTfull service, in this case a CoAP RESTfull service. The proposed architecture constitutes an economic solution as the evaluation of the data is done in the IoT edge network and only useful information is transferred to the cloud. This is the first step in the construction of a complete edge-computing environment in which

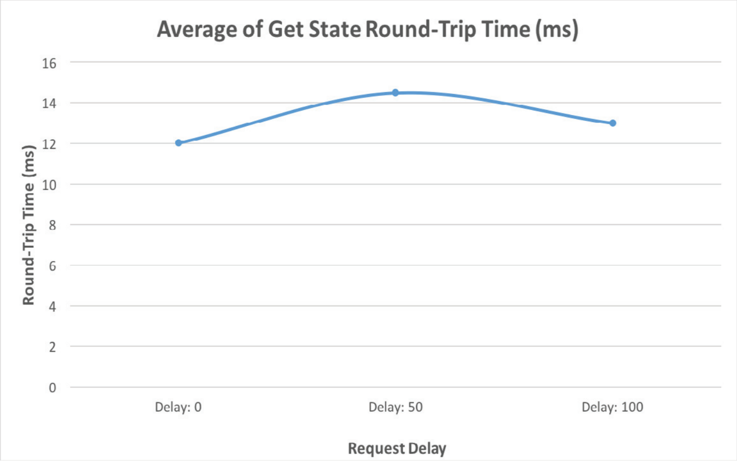

Fig. 9. One thousand of virtual clients. Virtual Resource Get-State Process Round Trip Time.

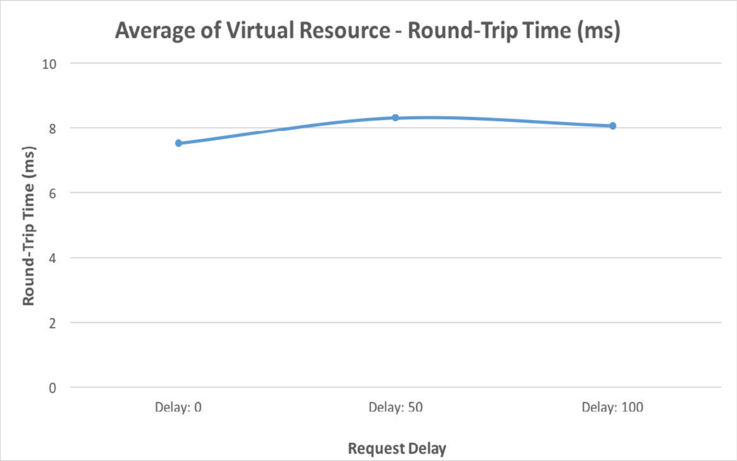

Fig. 10. One thousand of virtual clients. Virtual Resource Database Connection Round Trip Time.

virtual resources are introduced as a recommended option for management in diversified IoT networks. Thus, we can fulfil the vision of the Internet of Things.

As a future work, this research will introduce a tested with a large number of sensors and actuators to measure the overall application. Also, the work will include distributed virtual resources.

References

1. Ashton, K. That internet of things thing. *RFiD Journal 22*, 7 (2009), 97–114.
2. Atzori, L., Iera, A., and Morabito, G. The Internet of Things: A survey. *Computer Networks 54*, 15 (2010), 2787–2805.
3. Bandyopadhyay, S., and Bhattacharyya, A. Lightweight Internet protocols for web enablement of sensors using constrained gateway devices. *2013 International Conference on Computing, Networking and Communications, ICNC 2013* (2013), 334–340.
4. Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., and Ayyash, M. Internet of Things: A Survey on Enabling Technologies, Protocols and Applications. *IEEE Communications Surveys & Tutorials PP*, 99 (2015), 1–1.
5. Evdokimov, S., Fabian, B., Kunz, S., and Schoenemann, N. Comparison of discovery service architectures for the internet of things.
6. Castellani, A. P., Gheda, M., Bui, N., Rossi, M., and Zorzi, M. Web services for the internet of things through coap and exi. In *2011 IEEE International Conference on Communications Workshops (ICC)* (June 2011), pp. 1–6.
7. Nati, M., Gluhak, A., Abangar, H., and Headley, W. SmartCampus: A user-centric testbed for Internet of Things experimentSmartCampus: A user-centric testbed for Internet of Things experimentationation. *2013 16th International Symposium on Wireless Personal Multimedia*

Communications, WPMC 2013 - Co-located with Global Wireless Summit 2013 (2013).

8. Hakiri, A., Berthou, P., Gokhale, A., and Abdellatif, S. Publish/subscribe-enabled software defined networking for efficient and scalable iot communications. *Communications Magazine, IEEE 53*, 9 (2015), 48–54.
9. Gubbi, J., Buyya, R., Marusic, S., and Palaniswami, M. Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Generation Computer Systems 29*, 7 (sep 2013), 1645–1660.
10. Kovatsch, M., Mayer, S., and Ostermaier, B. Moving application logic from the firmware to the cloud: Towards the thin server architecture for the internet of things. In *Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2012 Sixth International Conference on* (2012), IEEE, pp. 751–756.
11. Mainetti, L., Mighali, V., and Patrono, L. A software architecture enabling the web of things. *Internet of Things Journal, IEEE 2*, 6 (2015), 445–454.