Using reactive paradigm with COAP protocol

Dalibor Fonović
University of Zagreb
Faculty of Electrical Engineering and Computing
Zagreb, Croatia
dalibor.fonovic@fer.hr

Mario Kušek
University of Zagreb
Faculty of Electrical Engineering and Computing
Zagreb, Croatia
mario.kusek@fer.hr

***Abstract*—The Constrained Application Protocol (COAP) is a lightweight communication protocol designed for efficient data exchange in resource-constrained environments and intended for use in IoT applications. However, the functionality of the COAP protocol is limited. By incorporating reactive programming concepts into COAP, its ability to manage dynamic and asynchronous communication can be improved to better meet the needs of modern IoT applications. The goal of this research is to investigate the integration of the reactive paradigm with the COAP protocol, which will be used in the dissertation as a mechanism to improve energy efficiency and reduce network congestion.**

***Index Terms*—Internet of Things, COAP, COAP Observe, Reactive programming, energy–efficency, network congestion**

I. Introduction

The Internet of Things (IoT) is a global network of devices capable of communicating and exchanging data with other devices and systems mainly via the Internet using specific communication protocols. The number of such devices that can be connected to the Internet due to progress in various technological areas (such as communication technologies, microelectronic circuits, sensors, embedded systems, smartphones etc.) is constantly growing. A constrained environment in the context of the IoT refers to a environment where devices have limitations in terms of computational resources, memory, processing power, energy supply and network connectivity (low bandwidth and high packet loss). These limitations differentiate constrained devices from traditional computing systems like desktop computers or servers. To enable communication and data exchange between devices in IoT several communication protocols are used. Some of the most popular protocols are HTTP, MQTT and COAP.

II. COAP protocol

The Constrained Application Protocol (CoAP) is a lightweight and efficient communication protocol designed for use in constrained environments such as the IoT devices. CoAP is similar to HTTP but it's specifically optimized for resource–constrained devices and low–bandwidth networks and is based on UDP. CoAP has a low overhead which reduces message size and thus consequently conserves bandwidth and energy usage.

CoAP supports four basic methods similar to HTTP. The core of the CoAP protocol is specified in RFC 7252 [1]. IoT devices often operate with limited battery life, so they use sleep mode, a power–saving state that IoT devices can enter to conserve energy. Because of the usage of UDP protocol there is no connection establishing and confirmation sending unlike TCP which makes better use of devices in sleep mode.

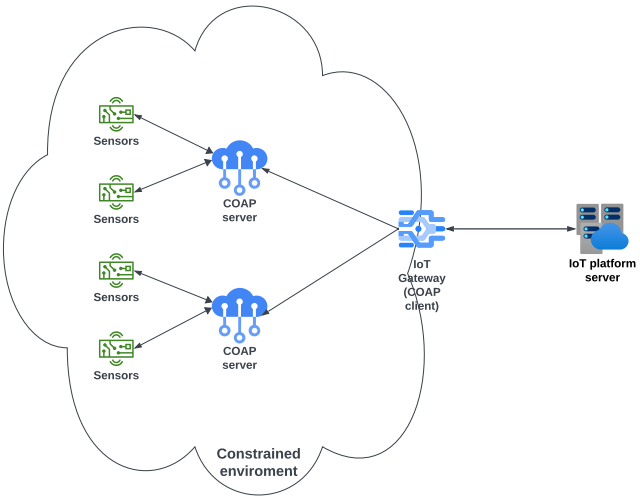

Fig. 1. Architecture of constrained environment using COAP communication protocol

An example of constrained IoT environment using CoAP protocol is shown in figure 1. CoAP server is IoT device that include sensors with limited resources (like microcontrollers). The IoT gateway is CoAP client and on the right side is IoT platform server. Because all mentioned above, we will focus on CoAP protocol in our research.

III. Reactive programming paradigm

Reactive programming is a programming paradigm that focuses on building systems that respond to changes and events in a flexible and efficient manner. It involves the use of asynchronous data streams, where data and events are treated as continuous flows that can be observed, reacted to, and manipulated by operators. Reactive programming is a declarative programming paradigm that is based on reactive paradigm.

A. Reactive application in IoT

Reactive programming in the context of the IoT refers to an approach that focuses on building IoT systems that are responsive to events and changes in their environment. IoT applications can react to incoming data, events, or conditions in a flexible and efficient manner.

Instead of relying on constant polling or periodic data exchanges, devices send and receive messages in response to significant changes or occurrences in their environment (i.e., if the temperature has dropped below 0°C).

IV. RELATED WORK

CoAP extensions in RFCs have only basic and limited functionality. What is missing in the current CoAP Observe extension [2] is the possibility to set certain requirements (i.e. the temperature has reached a threshold value).

Teklemariam et. al. [3] presented the concept of conditional observations as an extension to the CoAP protocol in general and the Observe option by implementing this functionality on a constrained device. The parameters for conditional observations are sent within the CoAP Option header. The obtained results showed that the conditional observations can be very useful extensions to the basic observe behavior in terms of application benefits and also regarding network efficiency considerations. Jung et. al. [5] are also extending CoAP for congestion control in streaming similar to video streming.

Troyer et. al. [4] presented an approach which facilitates and reduces complexity in the design of IoT systems by using reactive programming and event streams. The implementation is in Elixir running on RPI wich is not constraint devices that we are targeting. Peros et al. [6] presented an IoT extension that combines and operates on events based on their time context, which is important for time–sensitive IoT applications, using ReactiveX reactive programming framework. The implementation is in TypeScript wich is not applicable for running on microcontrollers.

V. RESEARCH CONCEPTUAL APPROACH

The focus of our research is in the part of constrained environment related to energy supply and network connectivity limitations. Our goal is to reduce the amount of transferred data and thereby improve energy consumption and reduce possible congestion in the constrained network, by using Reactive programming paradigm configured to send reactive parameters to the observed resource.

Our approach unlike research in [3] and [5] is not extending CoAP protocol. The proposed approach uses the existing CoAP initialization operation for creation of the Observe resource using the POST method like in [5]. As in the figure 2, the client first requests the creation of resource to the server with a message containing Reactive parameters. When the resource is successfully created, the receiver returns the URL of the generated resource via the 2.01 response message. One such example of reactive parameters can be sending a backpressure request in context of reactive programming from the client to the server. The server upon receiving the backpressure request, will adjust its sending rate based on the client's capacity to handle the data. By this mechanism we can control how much data will be sent over the network. In the future work we will investigate different reactive operators.

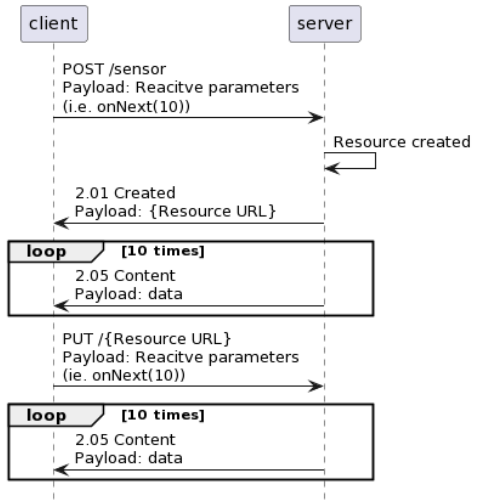

Fig. 2. CoAP Initialization with Reactive configuration

VI. CONSLUSION AND FUTURE WORKS

CoAP is designed to enable efficient and reliable communication between devices with limited resources, making it well-suited for various IoT and machine-to-machine applications. Reactive programming in IoT focuses on creating systems that can adapt and respond to dynamic conditions by using event-driven architecture, asynchronous processing, and real-time responsiveness. Combining COAP an reactive paradigm in IoT can be well-suited for building IoT applications that need to handle diverse and changing data sources and events. Our future work is to evaluate how the usage reactive functionalities/operators on a constrained device can improve energy consumption and reduce network congestion in the constrained environment (i.e. filter a certain condition met, like reaching the freezing temperature value) in specific use case.

REFERENCES

[1] Z. Shelby, K. Hartke, and C. Bormann, 'The Constrained Application Protocol (CoAP)', no. 7252. RFC Editor, Jun-2014.

[2] K. Hartke, 'Observing Resources in the Constrained Application Protocol (CoAP)', no. 7641. RFC Editor, Sep-2015.

[3] Teklemariam, G.K., Hoebeke, J., Moerman, I. et al. Facilitating the creation of IoT applications through conditional observations in CoAP. J Wireless Com Network 2013, 177 (2013). https://doi.org/10.1186/1687-1499-2013-177

[4] C. de Troyer, J. Nicolay and W. de Meuter, "Building IoT Systems Using Distributed First-Class Reactive Programming," 2018 IEEE International Conference on Cloud Computing Technology and Science (CloudCom), Nicosia, Cyprus, 2018, pp. 185-192, doi: 10.1109/CloudCom2018.2018.00045.

[5] Jung, J.-H.; Gohar, M.; Koh, S.-J. CoAP-Based Streaming Control for IoT Applications. Electronics 2020, 9, 1320. https://doi.org/10.3390/electronics9081320

[6] S. Peros and D. Hughes, "Reactive Programming Extensions for Time-Sensitive IoT Applications," 2021 17th International Conference on Distributed Computing in Sensor Systems (DCOSS), Pafos, Cyprus, 2021, pp. 244-251, doi: 10.1109/DCOSS52077.2021.00048.