Mittuniversitetet
MID SWEDEN UNIVERSITY

Campus Härnösand Universitetsbacken 1, SE-871 88. **Campus Sundsvall** Holmgatan 10, SE-851 70 Sundsvall.
Campus Östersund Kunskapens väg 8, SE-831 25 Östersund.
Phone: +46 (0)771 97 50 00, Fax: +46 (0)771 97 50 01.

Project report for course in Internet-of-Things protocols

Adam Temmel

MID SWEDEN UNIVERSITY
Department of Information Systems and Technology (IST)

Main field of study: Computer Engineering
Credits:
Semester, year: XX, YYYY
Supervisor: First name Surname
Examiner: First name Surname
Degree Programme: (optional)

Abstract

The abstract acts as a description of the reports contents. This allows for the possibility to have a quick review of the report and provides an overview of the whole report, i.e. contains everything from the objectives and methods to the results and conclusions. Examples: "The objective of this study has been to answer the question.... The study has been conducted with the aid of.... The study has shown that..." Do not mention anything that is not covered in the report. An abstract is written as one piece and the recommended length is 200-250 words. References to the report's text, sources or appendices are not allowed; the abstract should "stand on its own". Only use plain text, with no characters in italic or boldface, and no mathematical formulas. The abstract can be completed by the inclusion of keywords; this can ease the search for the report in the library databases.

Keywords: Human-computer-interaction, XML, Linux, Java.

Acknowledgements

Acknowledgements or Foreword (choose one of the heading alternatives) are not mandatory but can be applied if you as the writer wish to provide general information about your exam work or project work, educational program, institution, business, tutors and personal comments, i.e. thanks to any persons that may have helped you. Acknowledgements are to be placed on a separate page.

Table of Contents

Abbreviations

ACK	Acknowledge
AWGN	Additive White Gaussian Noise

1 Introduction

As Moore's Law foretold[1], we have been steadily increasing the ratio between computing power and metric area unit used to contain it. Computers are no longer contained to a single room, as they nowadays are able to fit in the palm of our hands. Another side effect of Moore's Law is our newfound ability to design communication protocols not only with respect to the computer, but to instead cater to the needs of developers, who might appreciate working with a protocol which is more comprehensible for humans instead of machines. While this is greatly appreciated for most use cases, there are a few outliers where these protocols are not feasible to use. One such case is the field of Internet-of-Things[2]. These devices are more tightly constrained in terms of resources when compared to your average personal computer, suggesting that they, depending on usage circumstances might require communication protocols designed for computers first and humans second.

1.1 Background and problem motivation

As IoT devices have been present for quite some time now, several different protocols targeting low-performance device communication have been drafted. Two of these are CoAP[3] and MQTT[4]. CoAP uses a REST-like model for device communication, whereas MQTT operates using a publish-subscribe model, meaning that they differ slightly in terms of what is and is not a suitable usage case for the device. Depending on the scenario, it might even be advantageous to join these two protocols when designing a larger system, depending on the specific needs of the individual components. It is not unreasonable to suggest that a system with more moving parts might very well be more complicated to author than a system with less moving parts. As such, it could be of interest to reconstruct such a situation in an attempt to later dissect it and discuss the ease of implementation for such a project, which is what this study tries to accomplish.

1.2 Overall aim

This project aims to reconstruct a rather basic (but scaleable) scenario between several components using various different communication protocols. In total, **five** different components are present within the system, with **three** different protocols being used, depending on the context. These protocols are the two aforementioned CoAP and MQTT, as well as the WebSocket protocol.

1.3 Concrete and verifiable goals / Detailed problem statement

The concrete and verifiable goals present for this project are as follows:

1. A working system consisting of at least 5 different components (including the end client) and 3 different protocols.
2. Partial implementations of the MQTT, CoAP and WebSocket protocol for usage within the project.
3. Working interactions between the author's protocol implementations as well as given library counterparts.
4. A benchmark able to assure the quality of the system.

1.4 Scope

Due to resource and simplicity constraints, the system will only be present on a single machine, which will detract some authenticity from the project, as a more authentic scenario would distribute the different components to different machines, depending on the use case(s) of what they are attempting to mimic.

1.5 Outline

Skriv det här din pajas!

Briefly describe the report's outline. "Chapter 2 describes…"

1.6 Contributions

Skriv det här din pajas!

Describe which parts of the work that you have conducted yourself, and which parts that you had help with i.e. carried out by colleagues. If the work is carried out in a group the report should then explain how the tasks were divided between authors. All co-authors should be credited in the work as a whole.

2 Theory

This chapter will present some underlying theory to understand the rest of the report.

2.1 TCP

The *Transmission Control Protocol* is a reliable protocol for data transmission over the internet[5]. The protocol features both checked and ordered transmission of data, leading it to being a widely used protocol for a variety of applications.

2.2 UDP

The *User Datagram Protocol* (sometimes referred to as the *Unreliable Datagram Protocol*) is a lightweight datagram-based protocol[6]. It is presented as an alternative to the TCP protocol for environments where the amount of available resources are too constrained for any potential usage of the TCP protocol. This is achieved by sacrificing the reliability features of TCP, meaning that UDP is less reliable as a consequence.

2.3 MQTT

The *MQTT* protocol is described as a lightweight publish/subscribe protocol designed with IoT devices in mind[4]. It is designed with a client-broker architecture in mind, meaning that clients with different agendas in mind all connect to a single broker which is responsible for distributing different messages. This distribution is handled by allowing clients to subscribe and publish messages to different topics, thusly achieving the aforementioned publish/subscribe functionality.

2.4 CoAP

The *Constrained Application Protocol* is described as a variant of HTTP suited for IoT devices[3]. All of the regular REST parameters are present within the protocol, with different resources being marked using URL paths to distinguish them. It can also operate using UDP as the underlying protocol, meaning that CoAP is well suited for IoT devices.

2.5 WebSockets

The *WebSocket* protocol was authored as alternative to HTPP for bidirectional communication between a server and client[7]. This includes instant messaging applications, games or other services which ideally operate without the need of reloading a webpage. It, like HTTP, uses TCP to provide a stable ground for the protocol.

3 Model

This chapter will discuss the model used to attain the goals of the project.

3.1 System overview

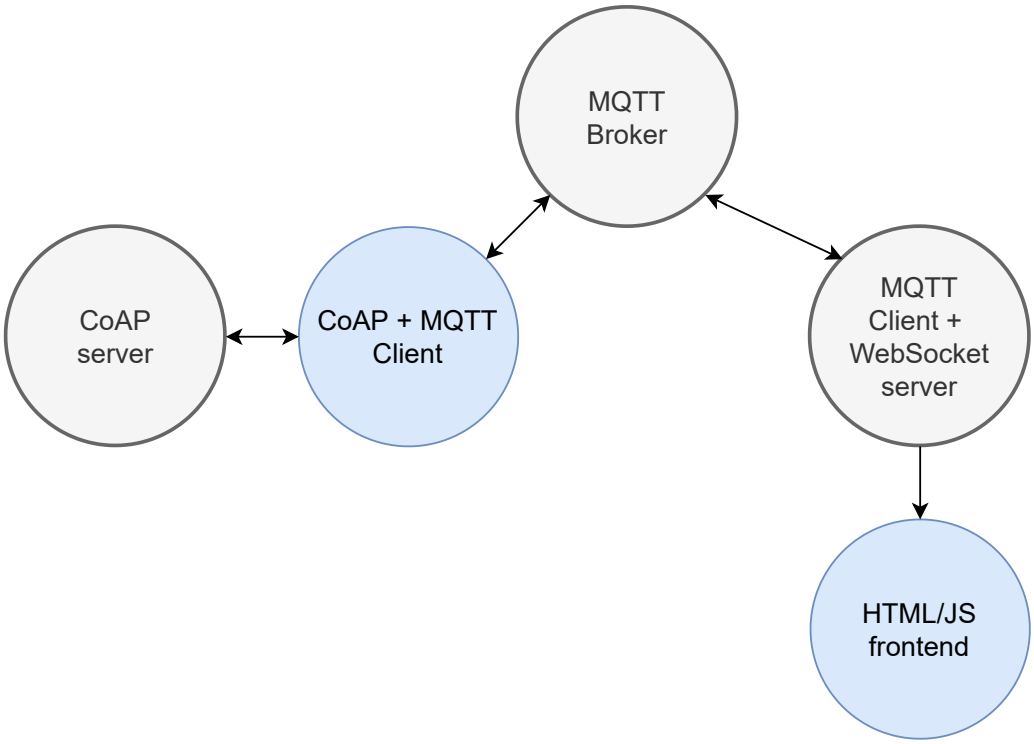

Figure 1: Overview of the system

Figure 1 shows a brief overview of the different components in the system. They are as follows:

3.1.1 CoAP server

A CoAP server responsible for providing current CPU and memory usage. This component should ideally symbolise some sort of IoT device which is of interest to monitor.

3.1.2 CoAP + MQTT Client

This component is responsible for picking up on MQTT based requests for current CPU/memory usage, then translating these requests into the

CoAP protocol and sending them to the CoAP server. Upon receiving a response from the CoAP server, this response is then translated back into a message able to be published to the MQTT broker.

3.1.3 MQTT Broker

The MQTT broker is responsible for managing all publications and subscriptions authored by the two MQTT clients. If either of the two clients wish to subscribe to a topic, this subscription will be registered and remembered by the broker. The next time a client wishes to publish something to this topic, this publication is retransmitted to all clients currently subscribed to this topic.

3.1.4 MQTT Client + WebSocket Server

This component works as a bridge between the MQTT broker and the end user frontend. It is meant to regularly publish requests for current CPU/memory utilization and subscribe to the topic in which the responses to these requests are published. Upon receiving a response, it is then translated into a WebSocket message which is sent to the frontend. This component should also be able to benchmark the RTT between publishing a request and receiving a response, to later enrich the message sent to the frontend with the measured time.

3.1.5 HTML/JS frontend

The frontend is a comparatively rather simple component, all things considered. It listens for published messages from the WebSocket server and updates three different graphs which plot CPU usage, memory usage and round trip time as a function of time. The frontend is not capable of sending messages in and of itself, but instead relies on the WebSocket server being able to pump out new information.

4 Design / Implementation

This chapter aims to discuss the implementation details of the system presented in *figure 1*.

4.1 CoAP server

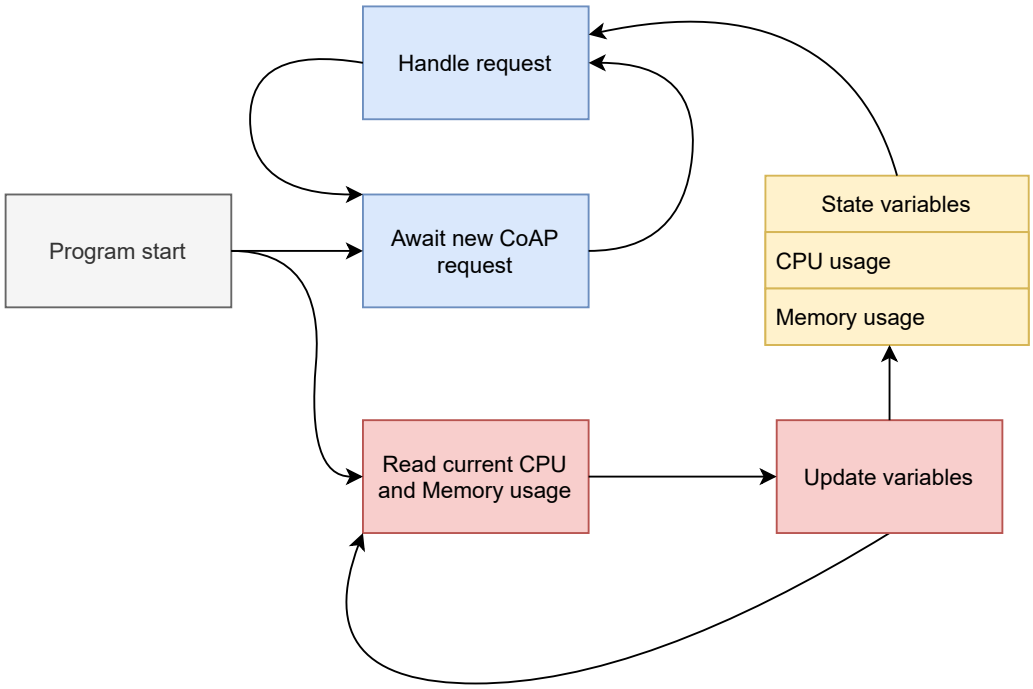

Figure 2: Flowchart depicting the duties and the program flow of the CoAP server.

4.2 CoAP + MQTT Client

Figure 2 shows the program flow of the CoAP server. The flow is separated into two threads marked as blue and red boxes in figure 2. There is also a yellow box, representing data shared between the threads. This data is locked behind mutex locks so as to not cause any race conditions between the red and blue thread.

The red thread is responsible for regularly reading the current CPU and memory usage from the system. It then tries to access the shared state variables in order to update them with the new information. Upon successfully updating the variables, the lock is released and the thread sleeps

for one second before attempting to read the CPU and memory usage once more.

The blue thread is responsible for performing all server-related duties of the system component. It listens for potential requests to the urls `/cpu` and `/mem`. Upon recieving a request, it tries to access the corresponding shared state variable, embeds it into a response, and sends the response. It then goes back into listening mode, awaiting the next request.

4.3 MQTT Broker

4.4 MQTT Client + WebSocket server

4.5 HTML/JS frontend

5 Results

The results chapter is included when you have produced a systematic study, i.e. an evaluation of a program that you have developed, which is required for C - and D-level diploma work. In the results chapter objective results of the empirical study are presented. Keep in mind that possible comments in this chapter should only be used for clarification. Your own views and subjective (personal) comments belong in the chapter conclusion/discussion.

Strive to present the results, for example measurement-, calculations- and/or the simulation result, in a form that is as lucid and easily understandable as possible. The results are preferably presented in diagrams or tables. Accounts of interviews can be summarised, but may include concrete examples supporting your work.

Extensive results, for example complete summaries of survey results, large tables and long mathematical deductions, are placed in the appendices.

6 Conclusions / Discussion

The conclusion/discussion (choose a heading) is a separate chapter in which the results are analysed and critically assessed. At this point your own conclusions, your subjective view, and explanations of the results are presented.

If this chapter is extensive it can be divided up into more chapters or subchapters i.e. one analysis or discussion chapter with explanations of and critical assessment of the results, a concluding chapter where the most important results and well supported conclusions are discussed and to sum it up a chapter with suggestions for further research in the same area. In this chapter it is of vital importance that a connection back to the aim of the survey is made and thus the purpose is pointed out in a summary and analysis of the results.

In this chapter you should also include answers to the following questions: What is the project's news value and its most vital contribution to the research or technology development? Have the project's goals been achieved? Has the task been accomplished? What is the answer to the opening problem formula? Was the result as expected? Are the conclusions general, or do they only apply during certain conditions? Discuss the importance of the choice of method and model for the results. Have new questions arisen due to the result?

The last question invites the possibility to offer proposals to others relevant research, i.e. proposal points for measures and recommendations, points for continued research or development for those wishing to build upon your work. In technical reports on behalf of companies, the recommended solution to a problem is presented at this stage and it is possible to offer a consequence analysis of the solution from both a technical and layman perspective, for example regarding environment, economy and changed work procedures. The chapter then contains recommended measures and proposals for further development or research, and thus to function as a basis for decision-making for the employer or client.

6.1 Ethical and Societal Discussion

You will need to include a discussion on ethics, societal impact, and considerations.

6.2 Future Work

You should also explain potential future work based on your work.

References

[1] Gordon E. Moore. "Cramming more components onto integrated circuits". In: *Electronics Magazine* (Apr. 19, 1965).

[2] Madakam S., R. Ramasway, and Tripathi S. "Internet of Things (IoT): A Literature Review". In: *Journal of Computer and Communications* (2015).

[3] *CoAP Homepage*. URL: `http://coap.technology/` (visited on 01/06/2022).

[4] *Mqtt Homepage*. URL: `https://mqtt.org/` (visited on 01/06/2022).

[5] Vinton G. Cerf and Robert E. Kahn. "A Protocol for Packet Network Intercommunication". In: *IEEE Transactions on Communications* (May 1974).

[6] *User Datagram Protocol*. URL: `https://www.ipv6.com/general/udp-user-datagram-protocol/` (visited on 01/06/2022).

[7] *The WebSocket Protocol draft*. URL: `https://datatracker.ietf.org/doc/html/rfc6455` (visited on 01/06/2022).

A Source Code

```c
#include <stdio.h>

int main() {
    printf("Hello, World!\n");
}
```