Relatório - Projeto Final de Arquitetura de Computadores

Projeto Arquiterura de Computadores

1st Bruno Martins Costa
Ciência da Computação
Universidade Federal de Catalão
Catalão, Brasil
Matrícula: 202101757

2nd João Pedro Pereira de Freitas
Ciência da Computação
Universidade Federal de Catalão
Catalão, Brasil
Matrícula: 202109749

3rd Miguel Affiune Carneiro
Ciência da Computação
Universidade Federal de Catalão
Catalão, Brasil
Matrícula: 202109750

***Abstract*—Este relatório propõe-se apresentar as características referentes ao projeto final dos alunos da turma de Arquitetura de Computadores (AC), cujo utiliza o protocolo MQTT e a conexão a internet para ligar uma lâmpada pelo uso do wifi através de um microcontrolador ESP8266, além de demonstrar a habilidade que os alunos adquiriram na criação e programação de circuitos eletricos, por meio de um trabalho estruturado e condizente com as formatações do modelo IEEE (The Institute of Electrical and Electronics Engineers).**

***Index Terms*—MQTT, Wifi, protocolo, microcontrolador, arduino, ESP.**

I. Introdução

O projeto desenvolvido pelo grupo de alunos tem como objetivo usar os conhecimentos adquiridos durante as aulas de Arquitetura de Computadores para criar um circuito utilizando o microcontrolador ESP8266. O grupo decidiu criar um circuito que utilizaria o protocolo MQTT (Message Queuing Telemetry Transport) e a conexão com a internet para ligar uma lâmpada através de um aplicativo no celular.

O grupo precisou fazer várias pesquisas para entender melhor como trabalhar com o MQTT (que será melhor exemplificado mais a frente nesse relatório), após várias análises de códigos que pudessem ser usados para trabalhar com o arduino, o grupo encontrou várias dificuldades, uma vez que os códigos eram um tanto complexos e se tratava de um conceito novo para todos no grupo. Porém, com um estudo mais aprofundado encontramos várias formas de trabalhar com esse protocolo, apenas tínhamos que escolher a mais simples de implementar e explicar.

Para criar o circuito os integrantes do grupo dividiram as despesas para comprar os componentes que seriam utilizados no circuito, esses componenetes incluiam um Relé[1], uma lâmpada led e um receptáculo para a lâmpada, os demais componentes usados no projeto foram fornecidos pelo professor da disciplina Tércio Alberto.

[1]Relé é um interruptor eletromecânico projetado por Michael Faraday na década de 1830, com inúmeras aplicações possíveis em comutação de contatos elétricos, servindo para ligar ou desligar dispositivos.

II. Trabalhos relacionados

Durante a preparação para este trabalho os integrantes do grupo realizaram diversas pesquisas sobre o assunto abordado, encontrando assim, diversos artigos e projetos que exemplificaram alguns dos principais tópicos, sendo alguns apenas curiosidades, e neste tópico será apresentado alguns deles juntamente com a interpretação e comentários de como esses trabalhos externos auxiliaram no desenvolvimento deste projeto.

A. Análise do Protocolo MQTT para Comunicação IoT através de um Cenário de Comunicação

O artigo é uma exemplificação de como e porque o protocolo MQTT foi criado e tem relação direta com a criação da própria internet. Basicamente, o protocolo MQTT foi desenvolvido pela IBM2, mas, atualmente, é um padrão aberto para comunicação entre dispositivos (OASIS, 2014), o que reforça o motivo de baseamento em tal protocolo para desenvolvimento deste projeto. Diferente de outros protocolos como, por exemplo, o HTTP, a arquitetura do MQTT é do tipo publicação e assinatura (publish-subscribers). Nessa arquitetura, o dispositivo é responsável por enviar (publish) as informações ao servidor, que opera como um intermediário (broker). Tendo conhecimento dos clientes que estão interessados nas informações enviadas (subscribers), o broker retransmite as informações recebidas. O MQTT é um protocolo situado na camada de aplicação da arquitetura TCP/IP. Ele define o modelo de operação entre os equipamentos, especificando os papéis de cada um, o formato das mensagens e a ordem entre elas. Adicionalmente, as funcionalidades da rede são providas pelas camadas imediatamente anteriores, com destaque para o protocolo TCP, para o protocolo IP (IETF, 1981a) e para as diferentes tecnologias de comunicação que estão na camada inferior dessa arquitetura (ethernet, wifi etc.).

Através do protocolo IP os dados transmitidos podem ser enviados em toda a Internet. Assim, da mesma forma que o HTTP, o MQTT pode ser utilizado com dispositivos ou brokers em qualquer lugar onde a Internet esteja disponível. Do ponto de vista da infraestrutura de comunicação, o MQTT pode ser implementado sobre qualquer tecnologia.

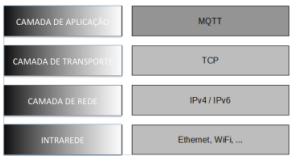

Fig. 1. Distribuição dos Protocolos nas Camadas

B. Internet das Coisas e os Principais Protocolos

O artigo dos estudantes do Instituto Federal de Sergipe (IFS) nos elucidou de como a IOT (Internet of Things/Internet das Coisas) utiliza os protocolos da internet, inclusive o MQTT. Segundo o artigo, a padronização da comunicação é um aspecto crucial para o desenvolvimento da Internet das Coisas. Nos últimos anos, diversos protocolos foram discutidos para atender aos requisitos das aplicações de IoT, sendo categorizados em D2D (device Data device) - Dispositivo a dispositivo, D2S (device Data server) - Dispositivo a Servidor e S2S (server Data server) - Servidor a Servidor. Os protocolos D2D conectam dispositivos diretamente entre si, enquanto os D2S coletam dados e os enviam para sistemas externos. Já os S2S são utilizados para gerenciar e integrar informações entre servidores de aplicação. Embora o IP seja a escolha óbvia para endereçamento na visão "Internet" do IoT, é necessário definir um protocolo de aplicação para a coleta de dados. O HTTP pode ser uma opção inicial, mas apresenta algumas limitações, como falta de qualidade de serviço, necessidade de polling[2]explícito e dificuldades na análise. Além disso, a segurança também é um problema que requer o auxílio de SSL/TLS. Por esses motivos, novos protocolos foram sugeridos, muitos deles baseados em um modelo publish-subscriber, para evitar polling, demandar menor largura de banda e resolver questões de conectividade. Entre os protocolos sugeridos, o MQTT e o CoAP se destacam como as principais opções.

O MQTT foi projetado para ser empregado em dispositivos de capacidade computacional reduzida, com baixa largura de banda e conectividade não garantida. A PDU (Protocol Data Unit) do protocolo MQTT é encapsulada pelo protocolo TCP, ou seja, o cabeçalho e os dados do MQTT são enviados na área de dados do TCP. Há uma versão do MQTT, denominada de MQTT-SN (MQTT Sensor Network), em que sua PDU é encapsulada pelo protocolo UDP, que, por sua vez, é encapsulada pelo IP ou pelo protocolo 6LowPAN. O MQTT prevê, ainda, diferentes garantias de entrega, denominado de QoS (Quality of Service), sendo o esse estabelecido de 0 a 2, respectivamente, com as semânticas: no máximo uma vez, pelo menos uma vez e exatamente uma vez.

Como visto, uma solução baseada em MQTT tem dois componentes: o broker e os clientes, que publicam e assinam tópicos. Há várias implementações em software livre para o broker (HiveMQ, ActiveMQ, RabbitMQ, CloudMQTT, Mosca...), nesse projeto foi utilizado o HiveMQ. Uma das implementações mais populares de broker MQTT é o Mosquitto2 e, para clientes, o Paho3.

[2]Polling, ou operação de polling, em ciência da computação, refere-se à amostragem ativa do status de um dispositivo externo por um programa cliente como uma atividade síncrona.

C. Aplicação do Arduino Uno em Sistemas de Automação Residencial

Neste projeto dos estudantes do Centro Universitário de Campo Real eles utilizam o Arduino Uno juntamente com a assistente Virtual Alexa da Amazon e o ESP8266, e alguns outros componentes secundários como relês e buzzers[3] para criar um sistema que realiza pequenas tarefas de automação no dia a dia de uma residência. As funções do sistema são o controle de iluminação, controle de temperatura da cama e o sistema de segurança. Direcionamos nosso foco para o controle de iluminação, ja que essa é a ideia central do nosso projeto. Basicamente a ideia deles para a iluminação segue o sistema a seguir:

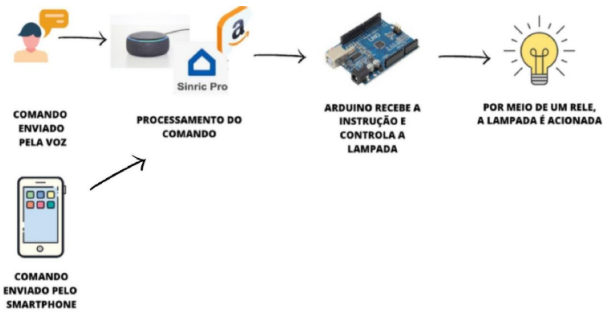

Fig. 2. Sistema de controle de iluminação através de um Assitente Virtual

Por um breve momento pensamos em utilizar um assistente virtual também, mas nenhum dos integrantes do projeto possuia um e seria um gasto muito alto obtê-lo. Sendo assim, decidimos que iriamos utilizar apenas um aplicativo no smartphone para realizar o controle da lâmpada.

III. Materiais e Métodos

A. Metodologia

A priori, é de suma importância ressaltar que para a validação dos conceitos apresentados no decorrer do projeto, tendo como exemplo o protocolo MQTT, o grupo buscou referências em projetos semelhantes e artigos. Sendo assim, o grupo reuniu-se através de videoconferêcias e reuniões para atualizações, estudo, definição, desenvolvimento do projeto e aplicação de metodologias ágeis como o Job-Rotation e o Kanban.

B. Ferramentas

Para a elaboração do projeto e execução de testes, o grupo inicialmente utilizou do simulador TinkerCad. Este sendo uma ferramenta online e gratuita para a elaboração de circuitos que, além de possibilitar a programação em linguagem Arduino e conter uma alta gama de componentes eletrônicos para se

[3]Dispositivo de sinalização de áudio

utilizar, possui a opção de obter-se uma vista esquemática do mesmo.

Após a fase inicial do projeto e teste dos componentes necessários pela plataforma virtual, os integrantes do grupo compraram três componentes que seriam necessários para a execução do circuito, sendo estes um relé, uma lâmpada led e um soquete para a mesma. Juntamente de tais componentes, utilizamos ferramentas básicas disponibilizadas pelo professor de Arquitetura de Computadores, Tércio Alberto. Estas sendo o próprio ESP8266, nossa placa de ensaio, os fios de condução, leds (posteriormente utilizados) e resistores.

IV. Descrição do Projeto

No projeto em questão, o desenvolvimento começou inicialmente na plataforma de simulação de circuitos online Tikercad. Dando continuidade ao projeto, devido a falta de compatibilidade de volts da nossa placa ESP8266 com a energia necessária para funcionamento do relé, foram listados os usos dos seguintes componentes:

- **Uma placa ESP8266;**
- **Dois resistores;**
- **Um LED amarelo e Um LED azul;**
- **Três fios condutores;**
- **Uma placa de ensaio;**

O código utilizado para criação e utilização do circuito foi escrito em linguagem de programação C++, direcionada as placas do Arduino Uno. Sendo inicialmente escrito na plataforma virtual de simulação de circuitos Tinkercad, e depois passada à própria IDE (Ambiente de Desenvolvimento Integrado) ”Arduino IDE”, onde utilizamos de base para conectar com o ESP8266 do grupo e testar e executar o objetivo do projeto.

O projeto tinha como objetivo simular uma versão de casa inteligente, onde por meio de um aplicativo de celular, o residente poderia acender as luzes da própria casa ou cômodo de preferência. Para isso utilizamos o protocolo MQTT de comunicação via Wifi.

Com o objetivo em mente e após os testes feitos no ambiente virtual, os integrantes do grupo se juntaram para comprar os componentes restantes para funcionamento do projeto. Como objetivo de acender as luzes de um cômodo de forma inteligente e prática, o primeiro item em questão foi a lâmpada led, um relé de 5V, o que gerou um conflito na compatibilidade da voltagem fornecida pelo nosso ESP8266, explicada futuramente, e um receptáculo para o encaixe da lampada e conexão com relé.

Testes constantemente feitos e para continuidade foi-se utilizado a ferramenta MQTT Box. Essa ferramenta foi utilizada inicialmente para estabelecer a conexão MQTT com um servidor publico e sua porta, que se comunicava com nosso projeto. Exemplificando a questão, essa ferramenta inicial foi o primeiro protótipo para desenvolvimento de testes antes da comunicação pelo smartphone com o aplicativo MQTT Dash.

V. Análise dos Resultados

A. Análise do Circuito

A priori, é importante destacar que a fim de reproduzir uma fiel versão de uma casa inteligente, o grupo desenvolveu no TinkerCad um esquemático utilizando da placa arduino UNO, um relé SPDT e uma lâmpada associada a uma fonte de energia. O esquemático não apenas auxiliou na montagem do circuito da primeira versão do projeto, mas também nos estudos em relação a como um módulo relé trabalha (Eletromagnetismo).

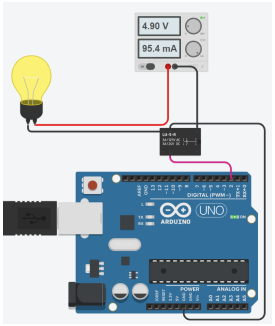

Fig. 3. Esquemático inicial - relé

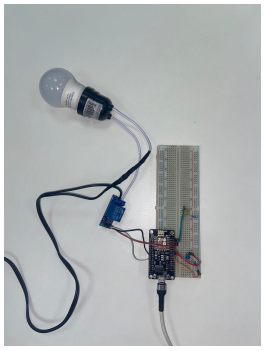

Fig. 4. Circuito inicial - Relé

Analogamente, quando aplicado fisicamente, o circuito não apresentou o funcionamento conforme esperado. Isso ocorreu devido ao ESP8266 possuir uma saída de alimentação de 3V, sendo que o módulo relé utilizado necessitava de 5V, voltagem que o Arduino UNO possui como saída. Apesar de momentaneamente ter funcionado corretamente, o módulo relé apresentava oscilações e a lâmpada ficava piscando, como se estivesse em ”meia fase”.

Tendo isso estabelecido, o grupo optou por uma versão simplificada do circuito, utilizando de dois LEDs como representação das lâmpadas a serem controladas pelo protocolo MQTT.

B. Análise do Código

Para codificar o projeto, o primeiro passo foi estudar a respeito das conexões MQTT com o ESP8266. Feito isso, descobrimos ser necessário a inclusão das bibliotecas

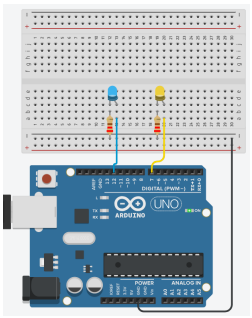

Fig. 5. Esquemático definitivo - LEDs

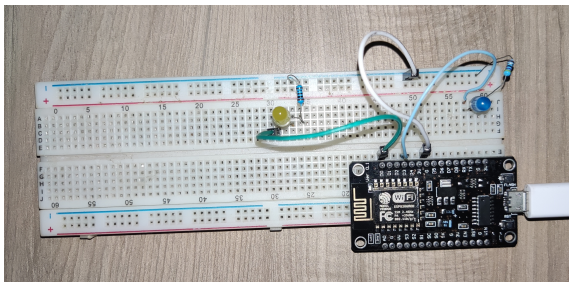

Fig. 6. Circuito definitivo - LEDs

ESP8266WiFi.h e PubSubClient.h, essas que permitem o microcontrolador conectar-se a uma rede Wi-fi local e a um cliente MQTT, incluindo a inscrição e publicação nos tópicos estabelecidos.

```
#include <ESP8266WiFi.h> // Inclui a biblioteca Wi-Fi
#include <PubSubClient.h> // Inclui a biblioteca MQTT

// Credenciais para se conectar ao Wi-fi
const char* ssid = ".";
const char* password = "nopass123";

// Credenciais do MQTT broker e topics
const char* mqtt_broker = "broker.hivemq.com";
const char* topic1 = "led/builtin/L1"; //Tópico LED 1
const char* topic2 = "led/builtin/L2"; //Tópico LED 2
const char* topic3 = "led/builtin/ALL"; //Tópico para acender todos os leds
const char* mqtt_username= "";
const char* mqtt_password = ""; //Usuário e senha vazios pois o broker utilizado é público

//Variável boobleana para indicar se o ESP se conectou ao broker
bool mqttStatus = 0;

WiFiClient espClient; // Cria um cliente Wifi
PubSubClient client(espClient); // Cria um cliente MQTT

//Protótipos
bool connectMQTT();
void callback(char *topic, byte *payload, unsigned int length);
```

Fig. 7. Credenciais para conexão ao Wi-Fi e MQTT

O código representado acima indica as credenciais necessárias para se conectar a rede Wi-fi e ao MQTT, incluindo SSID e password da rede, server (endereço do MQTT broker), tópicos que serão utilizados (neste caso foram definidos três tópicos, devido cada LED ser controlado individualmente ou em conjunto), e usuário e senha do broker, porém, devido o broker utilizado (HiveMQ) ser público esses campos ficaram vazios.

Ademais, após ter recebido as credenciais, o código acima cria um cliente Wi-fi e MQTT que permitirão a integração com o circuito. Importante notar a declaração da variável booleana MqttStatus que armazenará o retorno da função connectMQTT(), citada como protótipo junto a função callback() que recebe a mensagem publicada nos tópicos estabelecidos.

Na função setup, como representado na imagem abaixo, é iniciado a Serial com uma taxa de transferência de 9600 bits e definido os LEDs conectados as portas D0 e D3 como OUTPUT e forçando a iniciação dos mesmos desligados.

```
void setup() {
  Serial.begin(9600); // Inicializa a Serial

  pinMode(D0, OUTPUT); // Configura o Led1 como saida na porta D0
  pinMode(D3, OUTPUT); // Configura o Led2 como saida na porta D3

  digitalWrite(D0, LOW); // Força a inicialização do programa com o LED1 desligado
  digitalWrite(D3, LOW); // Força a inicialização do programa com o LED1 desligado


  WiFi.begin(ssid, password); // Conectar a rede Wi-fi
  while (WiFi.status() != WL_CONNECTED) { // While que confere se o ESP se conectou ao wi-fi
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("WiFi connected"); // Conectado com sucesso na rede wi-fi

  //Printa na serial o IP da rede conectada
  Serial.println(WiFi.localIP());

  mqttStatus = connectMQTT(); //Chama a função para se conectar ao MQTT e armazena o retorno da mesma
}
```

Fig. 8. Função setup()

Após a configuração dos LEDs e inicialização do monitor serial, o programa fará com que o ESP8266 conecte-se a rede Wi-fi cuja as credenciais foram passadas anteriormente e, após conectado, retornará no monitor serial o endereço IP da rede conectada. Por fim, a função setup() chama a função connectMQTT() que receberá armazenará na variável booleana MQTTStatus o valor do seu retorno, esse que influenciará diretamente na função loop().

```
void loop() {
  if (mqttStatus) { //caso mqttStatus tiver valor armazenado
    client.loop(); //execução do cliente mqtt definido
  }
}
```

Fig. 9. Função loop()

A função connectMQTT() iniciará configurando a conexão com o servidor do broker MQTT, passando como parâmetro o endereço definido anteriormente e a porta de conexão, após isso é configurado o callback. Caso todas as credenciais fornecidas no início do código forem válidas, retornará no monitor serial uma mensagem indicando êxito na conexão, se inscreverá nos tópicos definidos, publicará a mensagem "OFF" em ambos e retornará o valor 1. Importante dizer que será realizada uma tentativa de cinco vezes para se conectar ao broker, caso passe disso será retornado pela função o valor 0, indicando erro na conexão.

```
bool connectMQTT() {
  byte tentativa = 0;
  client.setServer(mqtt_broker, 1883); //cofnigurando o servidor
  client.setCallback(callback); //configurando o callback

  do {
    String client_id = "LAMP-"; //Define o nome do cliente MQTT
    client_id += String(WiFi.macAddress()); //Armazena na variável o MAC da rede

    if (client.connect(client_id.c_str(), mqtt_username, mqtt_password)) {
      Serial.println("Êxito na conexão"); //caso os parâmetros passados forem válidos

    } else {
      Serial.print("Falha ao conectar: ");
      Serial.print(client.state());
      Serial.println();
      Serial.print("Tentativa: ");
      Serial.println(tentativa);
      delay(2000);
    }
    tentativa++;
  } while (!client.connected() && tentativa < 5); //executa as tentativas de conexão enquanto a condição f

  if (tentativa < 5) {

    //Inscreve-se em um tópico e publica nele para obter um retorno de conexão
    client.publish(topic1,"OFF");
```

Fig. 10. Função connectMQTT()

Como dito anteriormente, a função loop depende do resultado retornado na função connectMQTT(), ou seja, caso tudo

ocorra em conformidade e seja retornado o valor 1, o cliente MQTT definido no início do código será executado.

Por fim, é necessário destacar a função callback() que receberá a mensagem publicada no tópico MQTT e ali será definido o que ocorrerá com elas. Neste caso, será informado através do monitor serial a mensagem que chegou, visto que a mesma é armazenada em um vetor chamado payload[] e, com base em condicionais, verificaremos em qual tópico a mensagem chegou, se a mesma for válida, acionará o(s) LED correspondente(s) ao tópico.

```
void callback(char* topic, byte* payload, unsigned int length) {

  //Informa através do monitor serial o tópico que a mensagem foi recebida e qual é a mensagem
  Serial.print("Mensagem chegou no tópico: ");
  Serial.println(topic);
  Serial.println("Mensagem: ");

  String message = ""; //armazenará a mensagem recebida
  for (int i=0; i<length; i++) {
    message += (char)payload[i]; //vetor que armazena a mensagem e armazena a variável
    Serial.print(message);
  }

  Serial.println();
  Serial.println("-----------------------------------------");

  // Verifica qual o tópico da mensagem recebida
  if (strcmp(topic, "led/builtin/L1") == 0) {
    // Controla o LED 1
    if (message == "1") {
      digitalWrite(D0, HIGH);
    } else {
      digitalWrite(D0, LOW);
    }
```

Fig. 11. Função callback()

C. Funcionamento

Para ilustrar, será demonstrado através de imagens o funcionamento do projeto em sua totalidade. Vale ressaltar que para publicar nos tópicos definidos, utilizamos o aplicativo MQTT Dash, disponível para Android, nele foi possível configurar botões como se fosse um aplicativo de Casa Inteligente, onde ao ser acionado publicará no tópico o valor 1.

Para configuração do aplicativo e dos botões é necessário incluir as informações do Broker MQTT como configurado no código do ESP8266, incluir um switch/button, fornecer um nome para o mesmo e indicar a mensagem e o tópico em que será publicada, podendo também definir um ícone e logo. Vale ressaltar que o Quality of Service (QoS) é definido na configuração do tópico em que será publicado, para o projeto deixamos no índice dois, pois é necessário que exatamente uma informação chegue ao tópico, devido se tratar de uma lâmpada, caso fosse um sensor, poderíamos deixar no íncide 0, que garante que praticamente uma mensagem chegará ao tópico, já que por segundo são várias informações mandadas.

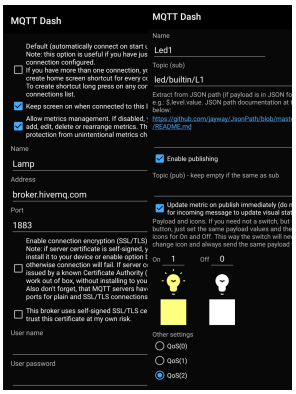

Fig. 12. Configuração do aplicativo MQTT Dash

Para acionamento do LED amarelo, basta clicar no botão nomeado como led1, caso esteja tudo OK, o mesmo ficará na cor amarelo e publicará no tópico referente o valor 1.

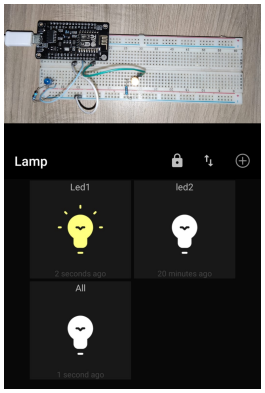

Fig. 13. Acionamento LED amarelo

Para acionamento do LED azul, basta clicar no botão nomeado como led2, caso esteja tudo OK, o mesmo ficará na cor azul e publicará no tópico referente o valor 1.

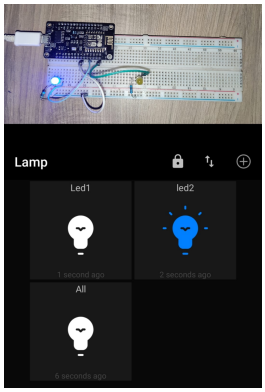

Fig. 14. Acionamento LED azul

Para acionamento de ambos os LEDS, basta clicar no botão nomeado como All, caso esteja tudo OK, o mesmo ficará na cor verde e publicará no tópico referente o valor 1.

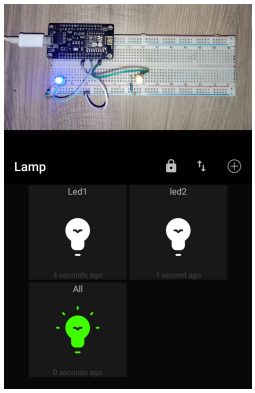

Fig. 15. Acionamento de ambos os LEDs

VI. Conclusão

Com a conexão MQTT estabelecida entre o ESP8266 e o smartphone, e a leitura dos dados de forma coesa, o objetivo da utilização desse sistema passa a ser o foco de observação e desenvolvimento.

Portanto, a simulação do funcionamento de lâmpadas em uma casa inteligênte, mesmo que de maneira alternativa, trouxe um leque maior de possibilidades ao grupo. Novas ideias relacionadas aos tipos de conexão Hardware e conhecimento sobre IOT (Internet of Things/Internet das Coisas), assim como funcionalidades do dia a dia que a evolução da tecnologia trouxe, de forma prática, aos seres humanos. Hoje é muito presente em grande empresas que fornecem esse serviço, como exemplo a Amazon com seus produtos de automação domiciliar, a famosa ”Alexa”, a qual trás todo esse conforto que a tecnologia permite.

Tendo em vista o projeto observado, conclui-se que, a partir dos resultados obtidos, cada LED e função é executada com sucesso, o que demonstra a confiabilidade do código implementado e a ressalta trazida no artigo ”Internet das Coisas e os Principais Protocolos”. Com a utilização correta da conexão MQTT, nos possibilitou uma menor largura de banda, evitar polling e resolver questões de conectividade como a facilidade na análise dos resultados, além de trazer uma fração do que a tecnologia prática pode ajudar na vida de cada, como uma simples simulação de casa inteligente.

References

[1] Análise do Protocolo MQTT para Comunicação IoT através de um Cenário de Comunicação - Wellington Nogueira Elizeu da Conceição, Romualdo Monteiro de Resende Costa Curso de Bacharelado em Sistemas de Informação –Centro de Ensino Superior de Juiz de Fora(CESJF) – Campus Academia 36016-000 – Juiz de Fora– MG– Brasil. Disponível em: 1688-4352-1-PB.pdf

[2] Revista Expressão Científica - Online - ISSN 2447-9209 — Volume II, Ano 02, Nº1 - 2017

[3] APLICAÇÃO DO ARDUINO UNO EM SISTEMAS DE AUTOMAÇÃO RESIDENCIAL]. Disponível em: https://www.revista.camporeal.edu.br/index.php/pi/article/view/519

[4] Bobsien, ESP8266 - envio e recebimento de dados vias MQTT. - Online - Disponível em: https://www.youtube.com/watch?v=EHTpWQ5s-No