Estação IoT para Monitoramento da Temperatura e Umidade do Interior de Veículos

Preprint · April 2018

DOI: 10.13140/RG.2.2.28236.51847

1 author:

Márcio Conceição da Silva
Instituto Federal de Educação, Ciência e Tecnologia da Bahia (IFBA)
2 PUBLICATIONS **0** CITATIONS

SEE PROFILE

Estação IoT para Monitoramento da Temperatura e Umidade do Interior de Veículos

Márcio C. da Silva – *Instituto Federal de Educação, Ciência e Tecnologia da Bahia*

***Resumo* — A internet das coisas é um campo recente, porém já oferece soluções de uso cotidiano através de *smartphones, tablets*, notebooks e outros *gadgets* que integram os objetos do cotidiano com a *internet*. Esta conexão pode ser realizada de diversas formas e sobre diversos protocolos, tanto para os sensores e atuadores no mundo físico, quanto para conexão e análise de dados na nuvem. Este artigo apresenta os resultados de uma estação para monitoramento da temperatura e umidade interna de carros através do módulo ESP8266 com o sensor DHT11. A análise dos dados será realizada pelo serviço online *ThingSpeak* com alerta para o usuário através da rede social *Twitter*.**

Palavras-chave* — internet of things, esp8266, dht-11. *Thingspeak, estação de medição.

I. Introdução

O desenvolvimento da internet e, principalmente, com a sua popularização na década de 90, tornou-se viável interligar equipamentos de uso cotidiano com a rede [2]. O conceito de internet das coisas é justamente esse, a conexão de equipamentos e objetos a rede, mais especificamente a internet. Em suma, as informações desejadas de certo objeto conectado, desde uma planta a um motor de grande porte, estarão disponíveis em qualquer local. A expressão *Internet of Things* (IoT) foi introduzida por Kevin Ashton, em uma conferência para a *Procter & Gamble* (P&G), em 1999, que despertou a atenção dos expectores e posteriormente tornou-se a designação para uma nova área da tecnologia [1].

As informações de temperatura e de umidade de um local são grandezas macroscópicas importantes do ambiente. Aplicações como o acompanhamento de e do clima de uma floresta podem auxiliar na preservação da mesma [6]. Em cidades densamente povoadas é possível utilizar um conjunto de sensores em um sistema IOT para detectar ilhas de calor e o período de temperaturas mais severas [5]. Assim medidas de controle podem ser aplicadas para minimizar estes impactos.

Na última década as pesquisas e artigos científicos relevantes cresceram de forma exponencial [7] na área de internet das coisas. Dentre estes, diversos artigos sobre de monitoramento de temperatura e umidade do ambiente com internet das coisas foram publicados. Destacam-se as pesquisas de medição de crescimento de plantas [6], estação de monitoramento de temperatura de *data center* [8], aplicação em agricultura de precisão [9], monitoramento e controle da qualidade do ar interno [10], nível de conforto do ambiente [7] e estação de monitoramento de poluição industrial [11].

Neste contexto, este artigo apresenta o desenvolvimento de um sistema de monitoramento de temperatura e umidade de veículos que possui semelhanças com os já citados, não obstante é uma aplicação diferente dos mesmos conceitos já estabelecidos.

II. Fundamentação Teórica

A. RFID

O sistema de RFID, *Radio Frequency Identification,* é composto por, basicamente, uma etiqueta e um leitor. As etiquetas são circuitos eletrônicos onde estão armazenadas informações sobre o equipamento e seu número único de identificação. E o leitor gera os sinais de radiofrequência para comunicação com a etiqueta [1]. O leitor pode ler, modificar e gravar novas informações na etiqueta. Uma das primeiras aplicações de uma plataforma de internet das coisas foi para o controle remoto de estoque. Neste sistema cada caixa do depósito recebe uma etiqueta gravada com os dados de interesse do produto como as dimensões, o peso, o tempo de estoque e a localização no armazém. Leitores distribuídos coletam os dados e os enviam para uma um sistema de controle através da rede local por conexão sem fio. O sistema de controle armazena as informações no banco de dados e recebe instruções do sistema de gerenciamento que direciona o conjunto de transportadoras de acordo com as informações nas etiquetas. A computação da nuvem permite o acesso as informações dos produtos de forma remota e análises como de tempo no estoque, taxas de envios e quantidade em estoque como é ilustrado na Fig. 1.

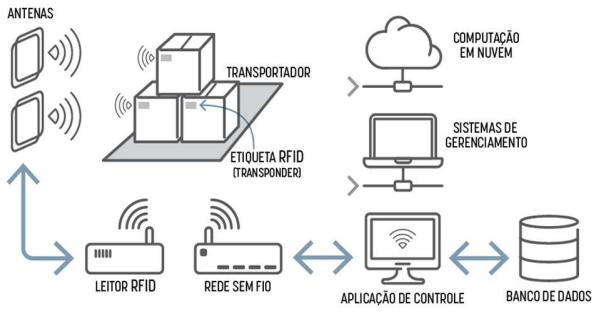

Fig. 1. Sistema de gerenciamento de estoque com RFID

B. Comunicação

A comunicação tem a função de determinar como os dados coletados pelos sensores, sistema RFID e outros dispositivos serão enviados fisicamente pela rede [1]. As tecnologias de rede se desenvolveram de forma rápida e ampla. Inicialmente com a internet o e protocolo TCP/IP que permitia a ligação através das "coisas via cabo". Posteriormente com o desenvolvimento das redes sem fio, particularmente o protocolo Wi-Fi. Porém, o grande impulsionador da integração das "coisas" foi o *smartphone*. Este dispositivo trouxe a mobilidade necessária para aceso fácil e desejável de informações em qualquer localidade.

Dispositivos como *notebooks* e *tablets* também, em uma escala menor, são responsáveis pela popularização da internet das coisas, justamente devido a sua portabilidade. Estas aplicações são possíveis graças ao desenvolvimento das redes móveis 2G/3G/4G.

> As redes de telefonia celular 2G/3G/4G também foram fundamentais para possibilitar a comunicação de dados dos diversos tipos de equipamentos móveis ou, ainda, aqueles cujo acesso a fios de comunicação era inviável. A comunicação de dados se tornou acessível a vários tipos de equipamentos e recursos, bem como reduziu o seu custo e o tempo de integração. (Oliveira, 2017, p. 5).

O desenvolvimento das tecnologias de comunicação em rede tornou a conexão à internet mais barata e por consequência mais acessível à população. Atualmente, conforme pesquisa do Centro de Estudos sobre as Tecnologias da Informação e Comunicação CETIC, 54% das residências brasileiras está conectada de alguma forma a internet, este percentual representa cerca de 36,7 milhões de lares conectados [3]. O CETIC é um órgão ligado a UNESCO (*United Nation Educational, Scientific and Cultural Organization*) para o desenvolvimento regional de tecnologia de informação na América Latina.

Se considerarmos o acesso em outros locais, como shoppings, universidades, restaurantes ou até mesmo na rua o número de brasileiros conectados é bem maior.

C. Padrões

Um dos grandes desafios para a implantação de soluções de IOT é a falta de comunicação entre dispositivos de fabricantes distintos, ou até, do mesmo fabricante. Esta ausência de interoperabilidade aumenta os preços dos produtos e serviços, dificultando a aceitação no mercado [1]. A integração de um smartphone a um automóvel só é possível graças à utilização do padrão *Bluetooh*, por exemplo.

O principal objetivo da rede em *internet of things* é a transmissão dos dados, dos objetos e do mundo físico, para análise na nuvem, a *analytics cloud*. O dispositivo de conexão entre as duas pontas é o *gateway*, que recebe os dados das "coisas" (segundo certo protocolo), converte essas informações em protocolo de rede para a transmissão de dados, como é exemplificado na Fig. 2.

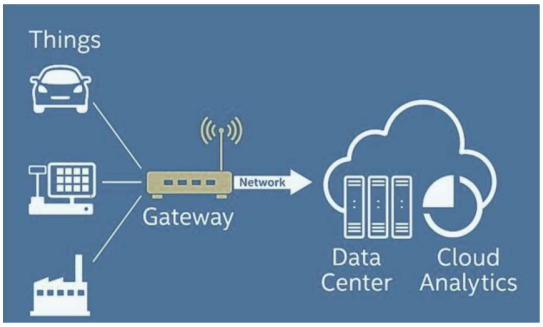

Fig. 2. Comunicação via *gateway*

A rede da Internet das coisas pode ser interpretada em três camadas: camada física, camada de rede e a camada de aplicação. Na camada de aplicação ocorre a interface entre as "coisas" e o *gateway*, que corresponde ao lado esquerdo da Fig. 2. Os protocolos desta camada podem ser divididos quanto ao alcance e a taxa de transmissão de dados:

1) *Protocolos de curto alcance e baixo volume de dados:* NFC, IEEE 802.15.4 e *ZigBee*.
2) *Protocolos de curto alcance e alto volume de dados:* *Bluetooh* e Wi-Fi;
3) *Protocolos de longo alcance e baixo volume de dados:* SIGFOX, LoRaWan, Narrowband e Nwave;
4) *Protocolos de longo alcance e alto volume de dados:* Redes de celulares 2G/3G/4G e LTE.

Já a camada de rede é responsável pela transmissão de dados do gateway para a nuvem. Os protocolos desta aplicação em geral são IPv4, IPv6 e para baixas potências o IETF IPv6 (6LoWPAN).

Na camada de aplicação é estabelecido o padrão de interação *machine-to-machine* M2M. Nesta camada o protocolo que regula a interação entre o dispositivo da nuvem (computador, *tablet*, IHM, *smartphone, data center*) e os sensores, atuares e objetos do mundo físico. O protocolo mais conhecido é o TCP/IP, padrão da internet atual, o mesmo utiliza uma estrutura cliente/servidor para comunicação de dados.

Outro protocolo que vem ganhado destaque é o *Message Queue Telemetry Transport*, MQTT, que tem como base o TCP/IP. Inicialmente o objetivo desde padrão era a supervisão e coleta de dados para sistemas do tipo SCADA (*Supervisory Control and Data Acquisition*). Em subestações este sistema é utilizado para conexão da Unidade de Aquisição de Dados e Controle (UAC) dos equipamentos e o sistema de controle.

O MQTT é um padrão aberto (*open-source*) desenvolvido pela IBM na década de 90 e opera com a filosofia *publish/subscribe* [4]. Os dados são enviados pelos *publishs* a um *broker* que gerencia o acesso dos dados aos clientes (*subscribes*). "Ou seja, ao invés de enviar mensagens para um conjunto pré-definido de destinatários, os remetentes publicam suas mensagens em um tópico específico no broker", (DIAS, 2016, p.111). A cada nova mensagem o gerenciador envia as mensagens aos destinatários que se inscreveram no tópico previamente conforme Fig. 3.

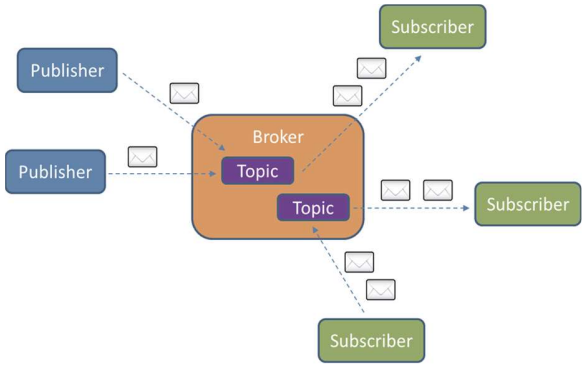

Fig. 3. Estrutura de comunicação do MQTT

Outro padrão da camada de aplicação é o *Constrained Application Protocol*, CoAP. Este protocolo é utilizado em redes restritas que usam dispositivos de baixo uso energia, como microcontroladores de 8 bits. Devido as suas características é utilizado em soluções *smart grid* que é a rede elétrica inteligente que utiliza tecnologia da informação para tornar o sistema mais econômico e confiável e ainda em automação residencial.

D. Segurança

Garantir a segurança das informações que transitam em uma rede de Internet das Coisas é um dos grandes desafios para aumento das aplicações desta área [1]. À medida que os padrões reduzem os custos de construção e ligam máquinas de fabricantes diferentes, os mesmos facilitam a exploração de falhas na rede por indivíduos mal-intencionados. As ameaças de segurança podem ser classificadas em quatros grandes grupos:

1) *Software:*
   Programas com erros e brechas na sua composição que podem causar falhas de segurança.
2) *Conectividade na rede:*
   Dispositivos projetados sem segurança de acesso e de protocolos podem causar falhas de segurança.
3) *Firmware:*
   A falta de atualização de firmware pode tornar os dispositivos alvos de uma invasão.
4) *Sistemas promíscuos:*
   Sistemas onde não existe controle interno de segurança e que não são dedicados a rede IOT são possíveis locais onde falhas de segurança podem ocorrer.

III. HARDWARE

O desenvolvimento de projeto de conexão dos componentes é verossímil ao desenvolvidos nos projetos de pesquisa citados anteriormente. Por meio de sensores e atuadores, sistemas de internet das coisas conectam o mundo real a aplicações virtuais que é exemplificado na Fig. 4. Linhas de pesquisa e desenvolvimento de monitoramento do ambiental seguem o mesmo esquema básico: um sensor de temperatura analógico como o LM35 ou de temperatura e umidade como o DHT-11/ DHT-22 acoplado a uma unidade de microprocessamento amigável como o ESP8266 ou a família de arduinos. A fonte de alimentação destes dispositivos pode ser efetuada através de ligação de circuito externo de corrente contínua, ligação com fontes chaveadas, tipicamente de celulares, conectadas por uma porta USB ou conexão pela porta USB do *notebook ou desktop*. Outro tipo de fornecimento de potência para o circuito é através de conjunto de pilhas AA, ou sistemas de baterias acopladas (*power bank*).

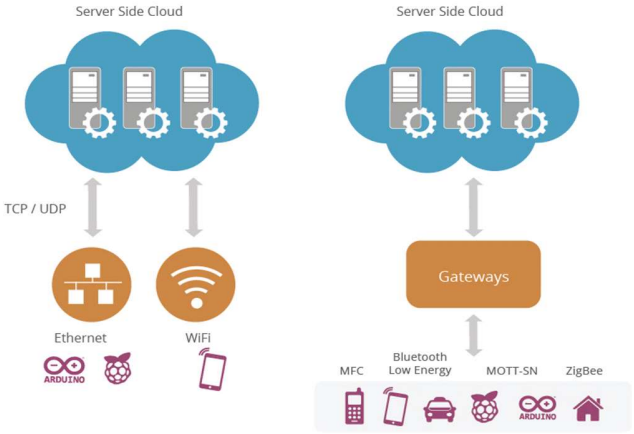

Fig. 4. Diagrama de sistemas IoT.

A. Sensor

Para medição de temperatura e umidade do local foi utilizado o sensor digital DHT-11. O mesmo é um circuito integrado que une um sensor capacitivo para medição de umidade relativa do ar e um termistor para medição de temperatura ambiente [7]. Este sensor possui um baixo custo de aquisição e apresenta precisão aceitável para ensaios não rigorosos. A tensão de I/O (input e output) é de 3 a 5 V com a corrente máxima de 2.5 mA. O diagrama de pinos é mostrado na Fig. 5.

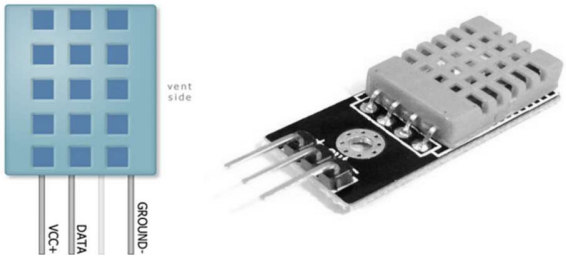

Fig. 5. Pinagem do DHT11

A faixa de medição de umidade relativa do ar que apresenta boa precisão e exatidão é de 20 a 90%. Já a faixa ótima de medição de temperatura de 0 a 50 ºC. A precisão do elemento é aceitável para fins didáticos, para umidade é de ± 5% UR e para a temperatura é de ± 2.0 ºC [9]

Devido a estas características este dispositivo é ideal para aplicação em sistemas de baixa potência como o uso de módulos como a família Arduino e os módulos ESP8266. A comunicação do sensor com estes dispositivos é digital o que para este projeto não representa erros significativos na faixa de operação. A tabela I agrupa a faixa de operação ideal do sensor.

TABELA I
CARACTERÍSTICAS ELÉTRICAS DO DHT-11

Característica	Mínimo	Máximo
Tensão de Entrada	3V	5V
Corrente de Entrada	0.5 mA	2.5 mA
Período de Amostragem	2 s	-
Medição de Umidade	20 %	90%
Medição de Temperatura	0 ºC	50 ºC

Inicialmente o ESP8266 envia um degrau negativo de cerca de 18 ms e em seguida é enviado em degrau positivo de curta duração (20µs a 40 µs), após isto o sensor envia os dados para o módulo [12] O sensor envia 5 pacotes de 8 bits (5 bytes). O primeiro byte informa a parte decimal da umidade, o segundo transmite a parte decimal da umidade, o terceiro pacote envia o valor inteiro da temperatura, o quarto byte informa os valores decimais da temperatura e por fim é enviado um byte de verificação de envio que é retratado na Fig. 6. Devido a este processo cada ciclo de leitura tem duração de 2 s, tempo aceitável tornando a medição praticamente em tempo real.

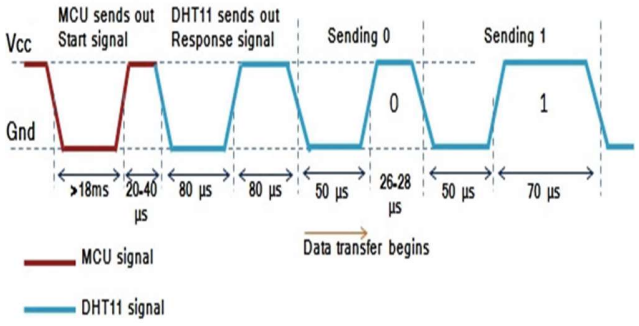

Fig. 6. Transferência de dados do DHT-11 para o ESP8266

B. *Módulo ESP8266*

O crescente mercado consumidor de produtos de internet das coisas a cada ano desperta a atenção de investidores. Somente no Brasil o mercado de IoT movimentou cerca de US$ 1.34 bi no ano de 2016 [13]. No mercado global a previsão de crescimento é exponencial. Segundo a Forbes a previsão é que até 2020 é de as transações de mercadorias relacionadas à internet das coisas alcancem US$ 457 bilhões, sendo que em 2016 este valor foi de aproximadamente 157 bilhões de dólares, o que representa uma perspectiva de uma taxa de crescimento anual de 28.5% [14].

Com este cenário promissor a empresa chinesa *Espressif Systems* lançou, em 2014, as primeiras versões do módulo ESP8266. Sendo seu público alvo desenvolvedores iniciantes de sistemas embarcados, principalmente, os que já utilizavam o Arduino.

A grande vantagem deste módulo está no baixo custo de aquisição, uma unidade básica é comercializada por US$ 3. Além disto o chip permite a integração do microprocessador a uma rede sobre o padrão IEEE 802.11 b/g/n (WIFI) sem a necessidade de um componente adicional, como é feito com o Arduino através das suas *shields*. Este fato torna o ESP8266 como um dispositivo de baixo custo, facilitando o acesso a acadêmicos e *makers* em geral.

A *Espressif Systems* tem demonstrado ser uma empresa amigável para os desenvolvedores disponibilizando bibliotecas, tutoriais, resoluções de problemas e fóruns de discursão. Porém as especificações do projeto do circuito integrado são segredo comercial.

O circuito é ofertado em diversas configurações que é uma estratégia da fornecedora para conquistar o mercado. A base para estas placas é o micro controlador ESP8266EX em que seu esquema é mostrado na Fig. 7:

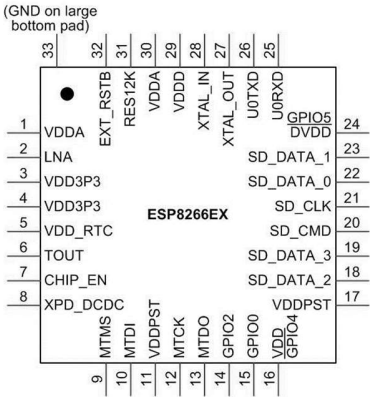

Fig. 7. Diagrama de pino do ESP8266EX

Este chip possui tamanho reduzido 5x5 mm, o que facilita a inserção em *protoboards*, contendo 32 pinos. A unidade de micro controle é Tensilica L106 de 32bits, cache de 64 kB e *clock* de 80MHz que pode ser aumentado até o dobro. A memória RAM pode variar de 50 a 96kB. Sendo assim possui poder de processamento superior ao do Arduino na sua versão Uno. A tensão de entrada do circuito é de 3.3 V em que nas versões de placa de desenvolvimento possuem um regulador de tensão permitindo conexão direta com USB. A zona de temperatura de trabalho pode oscilar de – 40 ºC até 125 ºC.

O ESP8266EX ainda apresenta um conversor analógico-digital de 10 bits, 16 pinos de entrada e saída GPIO com suporte a PWM, uma interface de periféricos seriais de 80 MHz, uma interface de integração dos circuitos entre os pinos GPIO14 e GPIO2 e uma UART (*Universal Asynchronous Receiver Transmitter*) que é utilizada para a conexão via WIFI com velocidade máxima de 4.5 Mbps [15].

É importe salientar que no uso direto do MCU é vital que a tensão de alimentação, V_{IN}, não seja superior a 3.6 V pois poderá haver danos ao componente, impossibilitando a conexão direta com sistema de 5V.

A família dos circuitos ESP8266 pode ser programada em linguagem LUA, *micropython* e em *wiring* através de uma biblioteca específica da IDE do Arduino. As placas diferem quanto ao tamanho, capacidade de conexão e de integração com os demais dispositivos. Abaixo é informado um breve resumo sobre as placas principais.

1) ESP-01

Este módulo é a versão mais básica do chip e possui apenas 2 pinos para conexão GPIO, como pode ser visto na Fig. 8. Este modelo compacto (24.8 x 14.3 mm) possui firmware atualizável com a possibilidade de ser regravado. Uma aplicação é de servidor web para acionamento das portas I/O. O esquema é mostrado na f

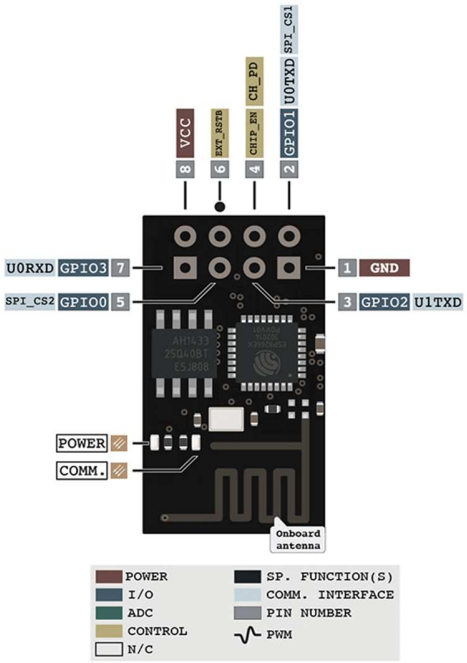

Fig. 8. Placa ESP-01

2) ESP-03

Este módulo possui mais conexões GPIO que a anterior e o seu diferencial é um pino para a conexão de uma antena externa o que faz aumentar o alcance conforme a Fig. 9. A posição dos pinos é mais bem arranjada que na versão ESP-01, já que na mesma é necessário um adaptador para a conexão com a protoboard.

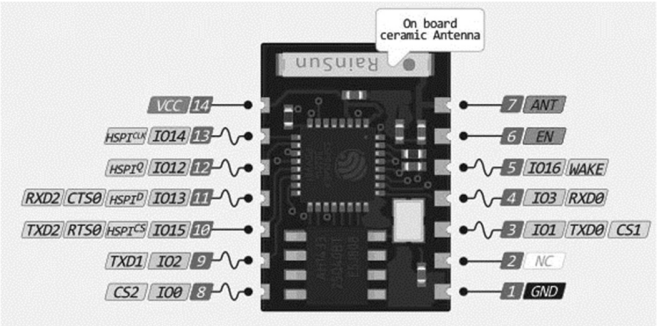

Fig. 9. Placa ESP-03

3) ESP-07

Este módulo não apresenta pinos soldados a base facilitando na aplicação de placas de circuito impresso. Mesmo de dimensões reduzidas (20 x 16 mm) apresenta 9 portas I/O sendo que estas operam I2C, SPI e PWM que é apresentado na Fig. 10. Possui ainda antena interna e conexão para antena externa, sendo amplamente utilizado em sistemas de automação residencial.

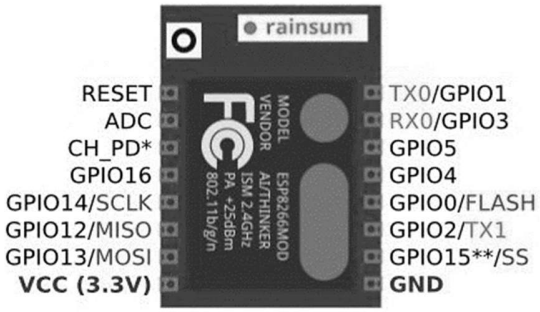

Fig. 10. ESP-07

4) ESP-12E

Este modelo difere muito pouco do anterior, a grande mudança é na antena. Enquanto no ESP-07 existe uma antena interna de cerâmica e um ponto para conexão de antena externa, neste dispositivo só existe uma antena interna que é de trilha conforme a Fig. 11. Este modelo é utilizado como base para o próximo dispositivo a ser citado.

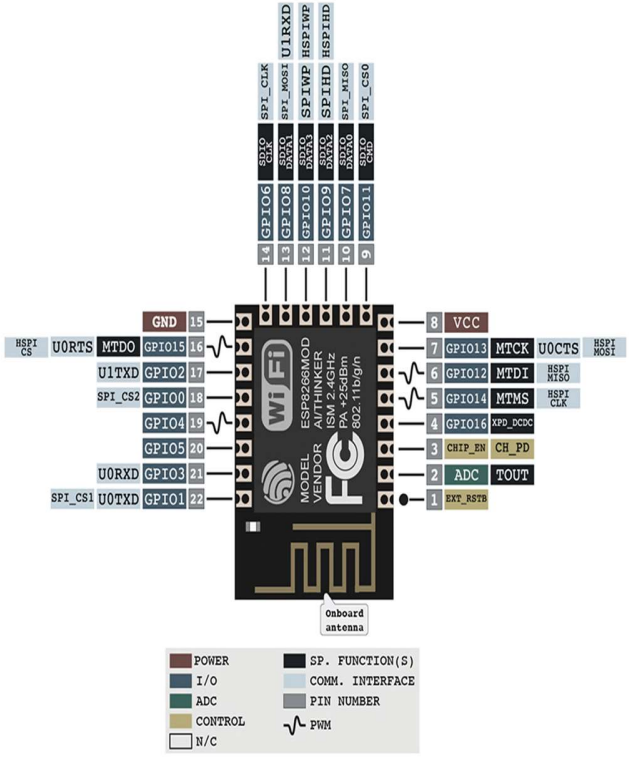

Fig. 11. ESP-12E

5) NODE MCU ESP-12E

Este modelo é um módulo de desenvolvimento completo pois conta com o ESP8266 ESP-12E, um conversor TLT-serial e regulador de tensão de alimentação. Esta opção dispensa o uso de um micro controlador externo e possui 11 pinos, Fig. 12, para a conexão que pode ser ligada diretamente na protoboard e fonte de tensão USB.

Por ser uma plataforma integrada e a facilidade de integração este módulo foi escolhido para realizar a conexão do sensor DHT-11 com a rede.

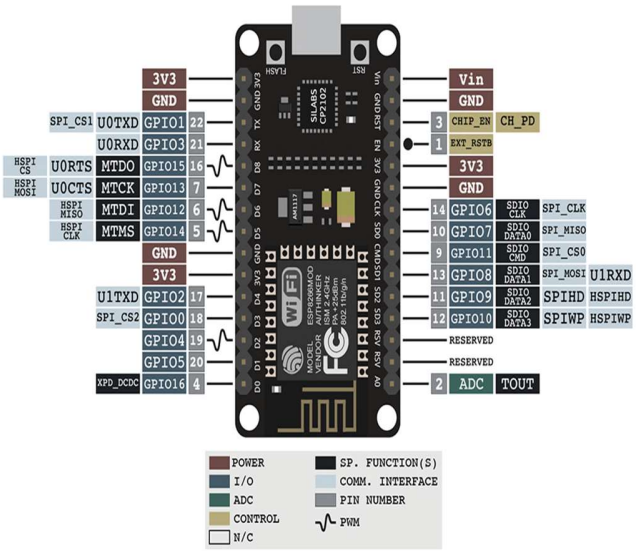

Fig. 12. NODE MCU ESP-12E

A tabela a seguir oferece um quadro comparativo das principais características dos circuitos integrados apresentados anteriormente.

TABELA II
QUADRO COMPARATIVO ESP8266

Produto	GPIO	Antena	Certificação
ESP-01	2	Impressa	-
ESP-03	7	Pino para conexão externa	-
ESP-07	9	Cerâmica e conexão para antena externa	FCC e CE
ESP-12E	9	Impressa	FCC e CE
NODE MCU 12-E	11	Impressa	FCC e CE

As certificações FCC e CE são selos que atestam a qualidade e possibilitam a venda de aparelhos de eletrônica e telecomunicações nos países que emitem estes selos. A agência reguladora americana *Federal Communications Commission*, FCC, fiscaliza as ações de radiofrequência, canais de rádio e tv, serviços de telefonia e tv à cabo. A esta organização também compete o controle de qualidade dos aparelhos elétricos e eletrônicos a serem vendidos nos Estados Unidos. Qualquer produto destas categorias fabricado ou não na América, deve possuir necessariamente o selo do FCC para ser comercializado.

A certificação do órgão de controle europeu é a CE, Conformité *Européenne*, que é a sigla em francês para conformidade europeia. De modo similar à autorização americana, este comitê fiscaliza e regula a venda de produtos eletroeletrônicos no bloco da União Europeia.

No Brasil este papel é desempenhado pela ANATEL que é a Agência Nacional de Telecomunicações. Os selos da agência americana, do conselho europeu e da agência reguladora brasileira são mostrados nas Fig. 13 a 15 sequencialmente.

Fig. 13. Selo da *Federal Communications Commission*

Fig. 14. Selo de Conformité *Européenne*

Fig. 15. Selo da Agência Nacional de Telecomunicações

IV. GERENCIAMENTO DE ENERGIA

O chip ESP8266EX desempenha 5 modos distintos de controle de energia. O primeiro modo é OFF em que o *Real Time Clock* (RTC) está desligado, que é o medidor do tempo presente do dispositivo. O segundo modo de operação é o DEEP_SLEEP em que o RTC permanece ativo enquanto o resto do chip está desligado. O próximo é o SLEEP, neste modo o oscilador de cristal (componente que gera um sinal elétrico em uma frequência específica) está desligado e o RTC ativo, sendo assim qualquer evento de inicialização deixa o chip ativo. O WAKEUP é o quarto modo de operação, o circuito integrado está ativo em operação normal. Já o quinto modo de operação é o ON, em que o chip opera com o oscilador a frequência nominal.

O consumo de energia é um dos parâmetros mais relevantes do projeto de sistema de baixa potência. Dispositivos com consumo elevado de potência podem inviabilizar projetos de baixo custo e/ou requerer componentes adicionais ou de maior capacidade como baterias e circuitos de transformação e regulação de tensão.

No módulo de desenvolvimento adotado a energia requerida depende da taxa de transmissão do sinal em rede sem fio, do tipo de padrão WIFI adotado (b, g ou n), da função desempenhada pelo dispositivo (transmissor ou receptor) e ainda do modo de operação de energia do dispositivo. Estes

parâmetros não foram disponibilizados pela fabricante, porém sites especializados na internet realizaram ensaios autônomos e estimaram estes parâmetros [16] como descrito na tabela III.

TABELA III
CONSUMO DE CORRENTE DO ESP8266

Modo de Operação	Consumo
Transmit 802.11b, CCK 1Mbps POUT=+19.5dBm	215 mA
Transmit 802.11b, CCK 11Mbps POUT=+18.5dBm	197 mA
Transmit 802.11g, OFDM 54Mbps POUT =+16dBm	145 mA
Transmit 802.11n, MCS7 POUT=+14dBm	135 mA
Receive 802.11b, packet length=1024 byte -80dBm	60 mA
Receive 802.11g, packet length=1024 byte -70dBm	60 mA
Receive 802.11n, packet length=1024 byte -65dBm	62 mA
WAKEUP	0.9 mA
DEEP_SLEEP	10 µA
SHUTDOWN	0.5 µA

Mesmo apresentando consumo elevado no modo de transmissão no padrão WIFI b, com cerca de 215 mA, o dispositivo pode ser considerado de baixa potência devido a sua corrente reduzida quando o dispositivo não está sendo utilizado. Para garantir certa autonomia para o dispositivo a fonte de alimentação escolhida foi de um conjunto acoplado de baterias, *power bank* que é ilustrada na Fig. 16. A bateria escolhida possui tensão de saída de 5V, com corrente máxima em 1 A e capacidade de 2200 mAh, o que disponibiliza cerca de 10 horas de funcionamento contínuo do aparelho com o modo de maior consumo.

Fig. 16. Modelo de *power bank* utilizado

A conexão com o circuito é feita através de cabo micro USB que é conectado ao ESP8266. É possível realizar o acompanhamento da tensão de alimentação online, através de linhas de código específicas, porém devido a autonomia oferecida pela bateria este comando não foi inserido.

V. MODELAGEM E DESENVOLVIMENTO DE HARDWARE

O sistema proposto neste artigo tem por finalidade o monitoramento da temperatura e umidade no interior de um veículo de forma remota. Um sensor captar os dados de temperatura e umidade do local. Acoplado ao sensor está o módulo ESP8266 que interpreta os dados obtidos do sensor e realiza a conexão com a rede WIFI local. Para o funcionamento adequado do projeto a rede local deve possuir conexão com a internet para transmitir os valores de temperatura e umidade para a plataforma online, que no projeto adotado é a *ThingSpeak*. No ambiente de desenvolvimento na nuvem é possível gerar gráficos para análise remota dos dados em tempo real e neste projeto também foi utilizado a alerta ao usuário através de mensagens enviada pela rede social *Twitter*. O diagrama do projeto é mostrado na Fig. 17.

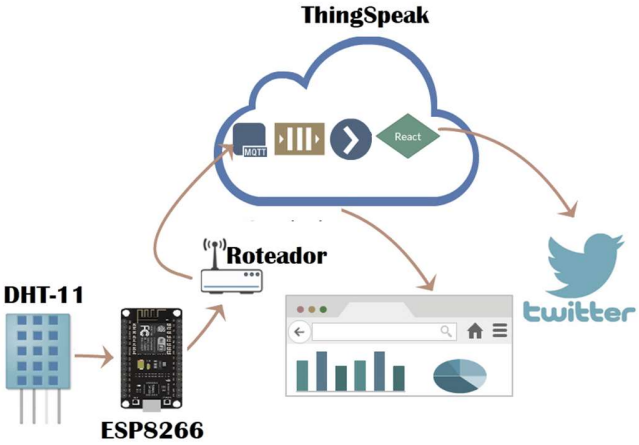

Fig.17. Diagrama conceitual do projeto

Para interligar de maneira adequada o sensor de temperatura e umidade a plataforma de desenvolvimento foi realizado um esquema de ligação com uso do software *Fritzing*. O *Fritzing* é um programa gratuito e *open-source* para modelagem de sistemas embarcados com foco em ligações com protoboard. A ligação dos entre os componentes deve seguir a união lógica de pinos conforme cada componente. No caso especifico a "pinagem" é descrita nas figuras 5 e 12, para o DHT-11 e Node MCU ESP8266 ESP-12E respectivamente. A conexão é a três fios. O fio vermelho conecta o terminal de tensão de saída (3.3 V) do micro controlador ao terminal de entrada de tensão do sensor. Analogamente o fio preto conecta o terminal terra (GND) dos dois componentes.

A transmissão de dados é feita com a interligação da porta GPIO 4, marcada como D2 no ESP8266, ao pino de dados do DHT-11, com o fio da cor laranja. A Fig.18 a seguir apresenta é a modelagem deste sistema no *Fritzing*.

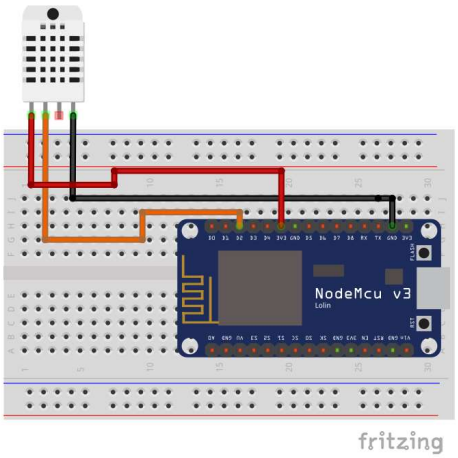

Fig. 18. Modelagem de ligação com o *Fritzing*

Após a modelagem foi realizada a montagem do esquema com o uso de uma protoboard de 400 pinos. É importante salientar que existem placas não oficiais disponíveis no mercado, com o tamanho maior que a original e não encaixam em apenas uma protoboard.

A ligação foi feita com jumpers seguindo o padrão de ligação e a coloração da figura 18. O sensor DHT-11 utilizado é da mesma fabricação que o do lado esquerdo da Fig. 5. Difere do esquema da imagem acima, pois o mesmo possui apenas 3 pinos. O pino central é destinado a conexão de dados e os demais seguem o esquema padrão (ou seja, foi removida o pino que não é conectado). A imagem (Fig. 19) retrata a montagem em bancada conforme projetado.

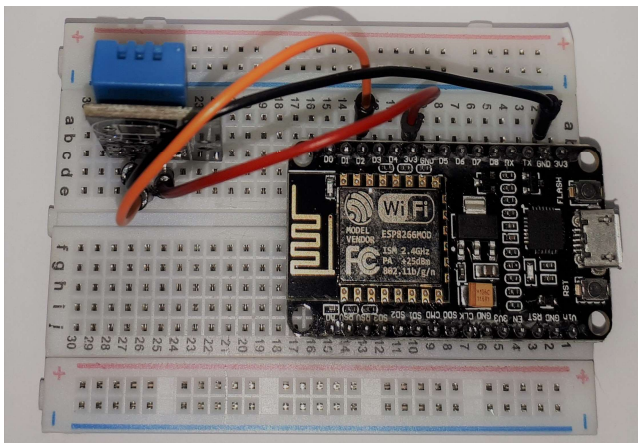

Fig. 19. Protótipo 1 – Montagem em bancada do modelo

A Fig. 20 mostra o protótipo acoplado a um carregador portátil de bateria (*power bank*) através de um cabo micro USB. O conjunto foi montado de forma a reduzir o espaço e evitar que as conexões se soltem.

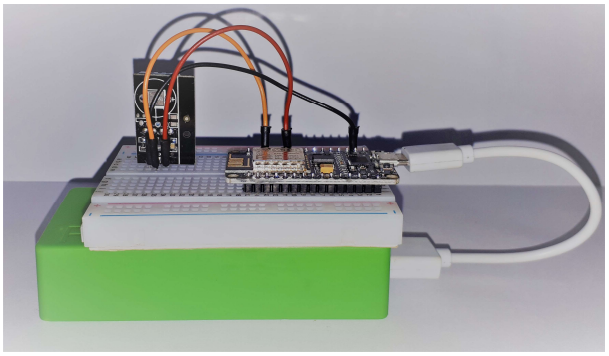

Fig. 20. Protótipo com bateria acoplada

Um dos grandes desafios de sistema baseado em prototipagem é presença frequente de conexões frouxas que causam maus contatos que podem danificar aos componentes. O protótipo da Fig. 20 apresentou diversas falhas na leitura no sensor devido o encaixe não está perfeito. Isto foi verificado em logs da porta serial no ambiente de desenvolvimento conforme a Fig. 21.

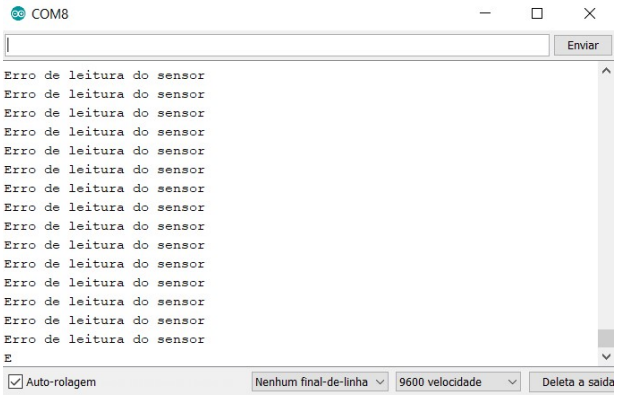

Fig. 21. Mensagem de erro na leitura do sensor

Para reduzir a perturbação gerada e viabilizar a medição da umidade e temperatura foi decidido a soldagem em placa fenolite dos componentes. A placa fenolite escolhida foi do tipo perfurada, como uma solução barata e simples. O circuito é equivalente ao da Fig. 18.

Novamente utilizamos o software *Fritzing* para modelar o esquema de soldagem e levantamento de materiais necessários para a confecção da placa o design projetado é ilustrado na Fig. 22.

Fig. 22. Modelagem para placa fenolite com o *Fritzing*

Com base da modelagem da Fig. 22 escolheu-se os materiais para a soldagem. Para a fixação da placa utilizou-se duas barras de 10 pinos para o ESP8266 e uma barra de 4 pino para o DHT-1, estas barras de pinos permitem uma boa fixação dos componentes sem a necessidade de soldar as peças diretamente na placa, facilitando a troca de peças e reaproveitamento das mesmas para outros projetos. Para efetuara a ligação entre as portas dos componentes foram feitas 3 trilhas com condutores de cobre nu e solda eletrônica.

As Fig. 23 e 24 mostram o dispositivo após soldagem. A Fig. 25 é mostrada a montagem do módulo NODE MCU e do sensor DHT-11 no dispositivo de fixação.

Após esta montagem foram novamente testados a comunicação entre os componentes. Desta vez foi possível realizar a leitura da temperatura e umidade praticamente sem erros como pode ser visto no log da porta serial do ambiente de desenvolvimento do programa como é ilustrado na Fig. 26.

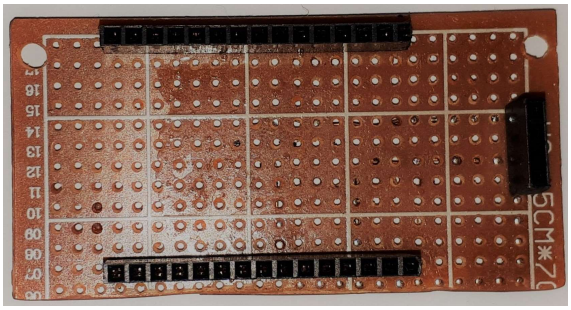

Fig. 23. Vista superior do dispositivo de fixação

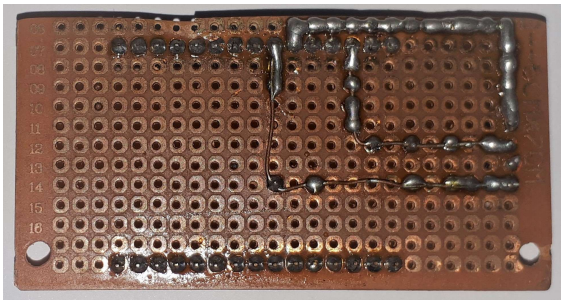

Fig. 24. Vista inferior do dispositivo de fixação

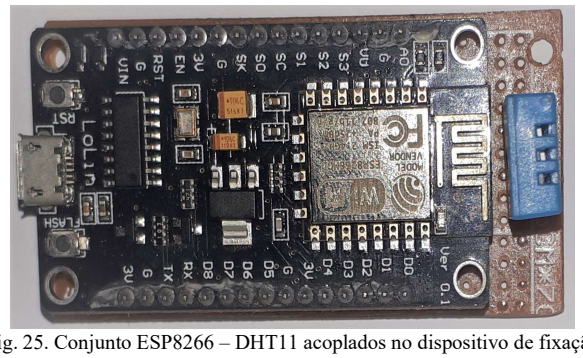

Fig. 25. Conjunto ESP8266 – DHT11 acoplados no dispositivo de fixação

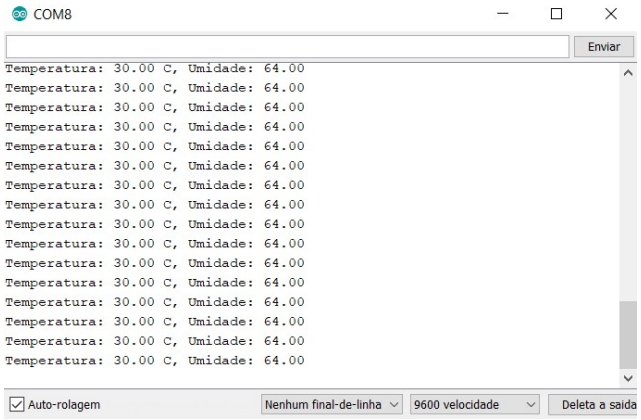

Fig. 26. Leituras bem-sucedidas na porta serial

VI. AMBIENTE DE DESENVOLVIMENTO

O módulo MCU ESP8266 possui um ambiente de programação em linguagem Lua, que é o SDK Lua. A linguagem Lua é uma ferramenta poderosa, de fácil semântica e vem ganhado força em projetos de automação residencial e prototipagem rápida. Esta linguagem foi resultado da pesquisa do Grupo de Tecnologia em Computação Gráfica da Pontifícia Universidade Católica do Rio de Janeiro sendo a primeira linguagem de impacto global desenvolvida fora do primeiro mundo. Porém o ambiente de programação tem se mostrado instável para diversos usuários o que prejudica na elaboração de projetos [17].

Felizmente a interface de desenvolvimento da plataforma arduino conta com a integração com o ESP8266, através de biblioteca específica. Assim é possível utilizar um ambiente de programação amigável (IDE Arduino) para programação do módulo MCU ESP. A linguagem de programação utilizada é a nativa do sistema arduino, o *wiring*. Na realidade o *Wiring* é um sistema de desenvolvimento de projetos de eletrônica que engloba tanto o hardware quanto o software. Este sistema foi desenvolvido no *Design Institute Ivrea*, na Itália e é voltado para projetos que não necessitem de alta performance, além de facilitar aos iniciantes em eletrônica. O hardware deu origem as placas Arduino e o software ficou com o nome do sistema original.

Wiring é uma linguagem baseada em C/C++ que com poucas linhas de comando é possível atuar sobre dispositivos eletrônicos sem o conhecimento profundo sobre o funcionamento dos mesmos. É uma linguagem de comandos simples e de documentação extensa.

A linguagem de programação *wiring* por ser baseada em C/C++ é uma linguagem compilada, ou seja, a mesma é executada diretamente pelo processador após a compilação o que é uma vantagem em termos de tempo de processamento e tamanho do programa em plataformas de sistemas embarcados.

A composição básica de um programa nesta linguagem é uma estrutura principal (void setup) e uma estrutura de repetição (void loop). Um programa desenvolvido nesta plataforma é denominado de *sketch* que em tradução livre

significa esboço. O ambiente de desenvolvimento para arduino (IDE) é de fácil utilização e conta com a tradução para diversos idiomas, inclusive para o português do Brasil. Nesta interface é possível a inclusão de bibliotecas (códigos de integração de dispositivos), visualização de dados através de um monitor serial, atualização de firmware, verificação do código e envio para o dispositivo de interesse. A estrutura básica de um *sketch* é mostrada na interface de desenvolvimento na Fig. 27.

Fig. 27. Sketch na IDE do arduino

VII. THINGSPEAK

Os dados obtidos das coisas do mundo físico são enviados para plataformas na internet para a análise e processamentos. Estes sites armazenam os dados e possuem uma interface para desenvolvimento de aplicações, *Application Programming Interface* (API). Estas interfaces são designadas de IoT *cloud*, onde é possível criar sistemas de interação com o usuário, direcionar as informações desejadas para outros dispositivos conectados na rede, analisar e armazenar os dados obtidos dos dispositivos de interface. Uma grande vantagem destes sistemas é a interoperabilidade com diversos dispositivos, graças a atuação em diversos protocolos de comunicação

Com o crescimento de sistema de internet das coisas a quantidade e a qualidade de sistemas de IoT *analytics* avançaram de igual forma. Atualmente, existem pelo menos 15 grandes plataformas de desenvolvimento na nuvem. Grandes empresas como a Microsoft, Amazon, Oracle, IBM possuem serviços de IoT *cloud* [7].

O *Thingspeak* é uma plataforma *open-source* de aplicações de internet das coisas que é baseada no protocolo TCP/IP (HTTP) e oferece suporte ao protocolo MQTT. Esta plataforma foi desenvolvida pela ioBridge em 2010 e posteriormente incorporada a *MathWorks* que é a empresa responsável pelo desenvolvimento do software Matlab. Esta aquisição permitiu o uso de ferramentas do Matlab nas aplicações sem a necessidade de uma licença. O foco do serviço deste site é a análise e monitoramento de dados numéricos.

O ThingSpeak possui integração com a rede social Twitter, por meio da aplicação *React.* A interface permite o envio de 8 campos de dados diferentes, além de um mapa com a localização da coisa/objeto.

Os valores transmitidos pelos dispositivos da interface física (ESP8266 e DHT11) são enviados a um canal previamente definido pelo usuário. A postagem das informações em cada canal é feita com uma chave de escrita a API key, que é um código alfanumérico exclusivo e que pode ser alterado. Este processo de autenticação garante que os dados enviados são direcionados para o canal correto. A figura abaixo mostra parte da interface do programa.

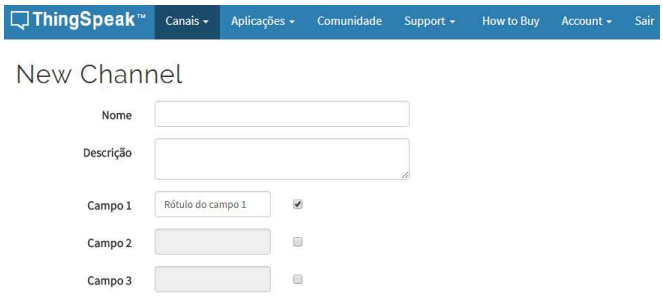

Fig. 28. Fragmento do ambiente de aplicações do ThingSpeak

VIII. CÓDIGO FONTE

O código utilizado foi separado conforme a sequência de imagens Fig. 29 – 38 para análise detalhada do mesmo. Inicialmente é inserida a biblioteca do Node MCU ESP8266 ESP-12E, como é descrito na linha 2, conforme a Fig. 29. Em seguida são inseridas as bibliotecas referentes ao sensor de umidade e temperatura DHT11 (linhas de 3 a 5). Estas bibliotecas estão disponíveis em repositórios online e são de fácil acesso.

```
1 // Bibliotecas
2 #include <ESP8266WiFi.h>//Biblioteca do ESP8266
3 #include <Adafruit_Sensor.h>
4 #include <DHT.h>
5 #include <DHT_U.h>
```

Fig. 29. Parte 1 do código fonte

Em seguida é realizada a inicialização do sensor de umidade e temperatura conforme Fig. 30. Na linha 7 é definido o tipo de sensor, pois a biblioteca utilizada também é compatível com o sensor DHT22. O comando seguinte define o pino de dados que conforme a fig. 18 é GPIO 4 (D2). A próxima linha inicializa o sensor conforme os parâmetros estabelecidos nos códigos anteriores.

```
6 //Inicialização do sensor de umidade e temperatura
7 #define DHTTYPE DHT11 //Estabelecendo o tipo do sensor
8 #define DHTPIN 4 //Pino de dados do ESP (GPIO4)
9 DHT dht(DHTPIN, DHTTYPE);
```

Fig. 30. Parte 2 do código fonte

O próximo fragmento do sketch define o modo de operação da placa WIFI do ESP8266 (Fig. 31). Neste caso o sistema servirá de cliente de uma conexão externa (com o roteador) do padrão IEEE 802.11.

```
10 //Definição do modo de operação do ESP8266
11 WiFiClient client;
```

Fig. 31. Parte 3 do código fonte

Em seguida são definidas variáveis para facilitar a programação, Fig. 32. Na linha 13 é inserida a chave de escrita (API *key*). Nas linhas 15 e 16 são declaradas a autenticação com a rede WIFI, com os dados do nome da rede e da senha da mesma.

```
12 //Variaveis auxiliares
13 String apikey = "E64FEYAHHUX0YS55";
14 const char* server = "api.thingspeak.com";
15 const char* ssid = "Y=X"; //Nome da rede a se conectar
16 const char* password = "H1N1INFLUENZA"; //Senha da rede
```

Fig. 32. Parte 4 do código fonte

A parte do programa apresentada nas Fig. 33 e 34 é a estrutura principal do programa. A mesma está contida entre os colchetes em aberto no final da linha 18 e fechado da linha 33. Na linha 19 é iniciada a porta de comunicação serial para possibilitar a obtenção de dados de forma local, já que a IDE de trabalho possui um monitor de porta serial que é definida pela saída USB do notebook que está conectada ao ESP8266.

A linha a seguir realiza a conexão com a rede WIFI com os parâmetros definidos previamente. Um laço de repetição é criado entres os comandos 22 a 25. O objetivo desta parte é a exibição no monitor serial de "..." caso a conexão com a rede local ainda não estiver concluída.

```
17 //Estrutura principal
18 void setup(){
19   Serial.begin(9600);
20   WiFi.begin(ssid, password); //Inicia a configuração de WIFI
21 //Aguardar a conexao com o roteador
22   while(WiFi.status() != WL_CONNECTED){
23     delay(500);
24     Serial.print("...");
25 }
```

Fig. 33. Parte 5 do código fonte

Após a conexão são exibidas as informações no monitor serial se a rede está conectada, o nome da mesma e o número IP atribuído como exemplificado na Fig. 39. Ainda no interior da estrutura principal o sensor DHT11 é iniciado de fato conforme linha 26.

```
26 dht.begin(); //Inicia sensor
27 //Logs da conexao de rede na porta serial
28   Serial.println("");
29   Serial.print("Conectado a rede: ");
30   Serial.println(ssid);
31   Serial.print("IP: ");
32   Serial.println(WiFi.localIP());
33 }
```

Fig. 34. Parte 6 do código fonte

A estrutura seguinte é a repetição principal, *void loop*, que está contida do colchete em aberto da linha 34 ao colchete fechado da linha 68. Este laço será repetido enquanto sistema estiver em operação. As instruções 35 e 36 armazenam a medição de umidade e temperatura, nas variáveis de ponto flutuante (números fracionários) umidade e temperatura. O bloco condicional, nas linhas 38 a 41, realiza a verificação se há de erro na leitura dos dados e informa a mensagem: "Erro de leitura do sensor" ao usuário, como mostrado na fig. 21.

```
34 void loop(){
35   float umidade = dht.readHumidity(); //Leitura da umidade
36   float temperatura = dht.readTemperature();//Leitura da temperatura
37    //Verificacao de erro de leitura
38  if(isnan(umidade)|| isnan(temperatura)){
39       Serial.println("Erro de leitura do sensor");
40  return;
41  }
```

Fig. 35. Parte 7 do código fonte

O segundo bloco condicional, linhas 42 a 60, realiza a postagem da temperatura e umidade no *ThingSpeak* através do protocolo TCP/IP. Na linha 43 é realiza a conexão com host através da porta 80. As linhas de comando 45 e 47 definem os campos de envio para o canal online, em que no campo 1 são enviadas as informações de temperatura e no campo 2 como mostra a Fig. 36.

```
42 //Conexao TCP para envio de dados
43 if (client.connect(server,80)){
44   String postStr = apikey;
45          postStr +="&amp;field1=";
46          postStr += String(temperatura);
47          postStr += "&amp;field2=";
48          postStr += String(umidade);
49          postStr += "\r\n\r\n";
50
```

Fig. 36. Parte 8 do código fonte

Caso a conexão seja efetiva é enviada uma solicitação ao servidor para fornecer as informações desejadas (linhas 51 a 59). Caso não seja possível à conexão é enviado um comando de encerramento na linha 61 conforme a Fig. 37.

```
    client.print("POST /update HTTP/1.1\n");
    client.print("Host: api.thingspeak.com\n");
    client.print("Connection: close\n");
    client.print("X-THINGSPEAKAPIKEY: "+apikey+"\n");
    client.print("Content-Type: application/x-www-form-urlencoded\n");
    client.print("Content-Length: ");
    client.print(postStr.length());
    client.print("\n\n");
    client.print(postStr);
}
client.stop();
```

Fig. 37. Parte 9 do código fonte

Por fim, para acompanhamento na tela do computador, é criado um *log* que informa os valores medidos de temperatura e umidade no monitor serial, Fig. 38. O monitor serial também apresenta o status da conexão como citado anteriormente e demonstrado na Fig. 39.

```
//Check de dados com a porta serial
    Serial.print("Temperatura: ");
    Serial.print(temperatura);
    Serial.print(" C, Umidade: ");
    Serial.println(umidade);

}
```

Fig. 38. Parte final do código fonte.

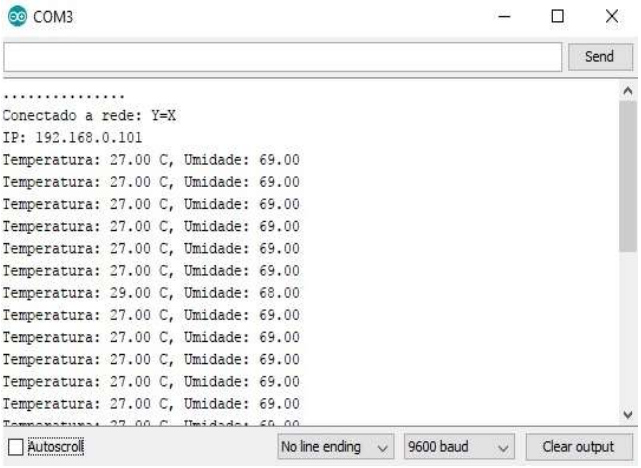

Fig. 39. Logs na porta serial

IX. Considerações Ambientais

Salvador no estado da Bahia, local em que esta pesquisa foi realizada, é uma cidade litorânea e possui valores elevados de umidade relativa do ar e de temperatura ambiente. Isto se deve ao fato da sua localização geográfica no nordeste do Brasil, região próxima a linha do equador. A média da umidade relativa do ar é superior a 80%. A temperatura média anual oscila entre 21 a 31 º C. Nas estações mais quentes a máxima pode atingir 37 º C conforme a carta climatológica do período de 1981 a 2010, do Instituto Nacional de Meteorologia, INEMET [18]. Nos dias em que a pesquisa foi aplicada à máxima foi de 34º C e 70% UR média.

X. Twitter

O Twitter é um site de interação social que conta com mais de 400 milhões de usuários ativos. Cada mensagem individual enviada pelos usuários pode conter até 280 caracteres, sendo categorizado como serviço de microblogging (textos curtos). Estas mensagens são denominadas de “tweets”. A interação entre os participantes é feita por um esquema de assinatura da informação, onde cada usuário “segue” (*follow*) os perfis de seu interesse e recebe a atualização em tempo real dos tweets postados nas contas seguidas [20]. Esta rede social possui integração com plataformas de desenvolvimento de internet das coisas e com outras redes sociais.

O ThingSpeak possui a integração com esta rede social através da aplicação REACT conforme a Fig. 40. Nesta aba é possível configurar a mensagem que será postada a partir da uma condição pré-estabelecida. A condição é uma comparação numérica como maior que, menor que, igual a maior ou igual a menor ou igual a e diferente de determinado parâmetro associada a certo campo de dados. A frequência de verificação da condição e envio de tweets pode ser configurada cada dado inserido ou a cada 10, 30 ou 60 minutos. Para tanto é necessário vincular uma conta de no microblog nas preferências do REACT e escrever qual o texto padrão a ser enviado.

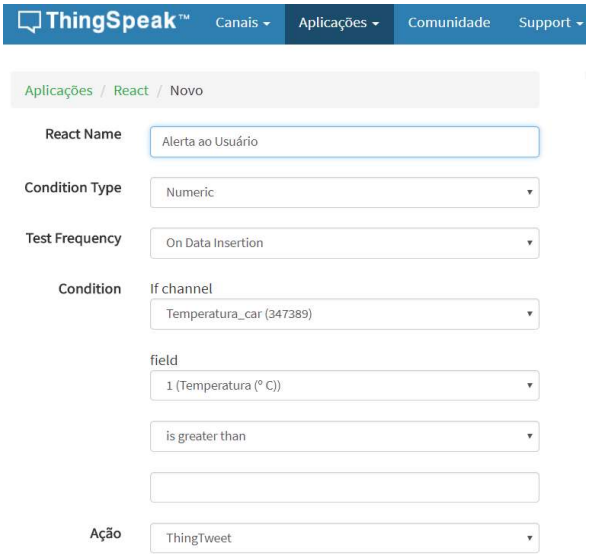

Fig. 40. Aplicação REACT do ThingSpeak

XI. Resultados e Discussão

A estação de monitoramento desenvolvida (Fig. 20 e 25) foi instalada no interior do veículo sobre o painel central. O carro estava estacionado em um local com exposição direta ao sol. Para amplificação dos resultados foi escolhido o horário de sol a pino em um dia claro, entre 12:00 h e 13:00 h.

O veículo em questão estava sem protetor solar de para-brisas. O teste inicial do sistema consistiu em conectar o conjunto de medição e observar através do monitor serial da IDE à conexão com a rede local e a leitura de dados pelo

ESP8266, Fig. 39. O ponto em que o carro estava localizado estava a 8 m do roteador.

A avaliação inicial indicou valores elevados de temperatura e valores muito baixos de umidade relativa do ar, ambos fora da faixa ideal de medição do sensor DHT-11. Mudou-se a posição do conjunto de medição para sobre o painel central, sobre o assento do motorista e sobre o banco traseiro, permanecendo por cerca de 5 min em cada posição.

Os dados foram enviados ao canal de ID 347389. Com auxílio da ferramenta MATLAB *analysis*, disponível na interface ThingSpeak foram calculadas as médias no intervalo, como exemplificado na Fig. 41.

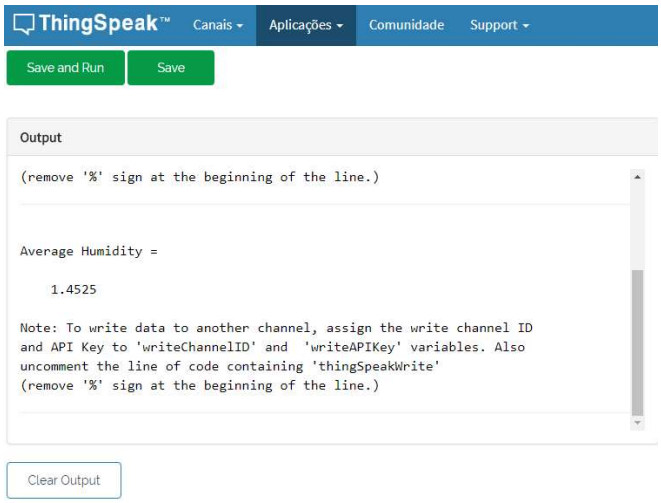

Fig. 41. Exemplo do cálculo da umidade relativa média com a aplicação MATLAB analysis

As variações de UR (umidade relativa do ar) com a posição foram pequenas com a maior umidade na parte traseira do veiculo. Porém a amostragem não é confiável, pois os valores obtidos estão abaixo da faixa de medição aceitável. Já a temperatura oscilou conforme a região amostrada. Acima do painel central a média obtida foi de 58,7 º C, que representa cerca de 18 º C a mais que a região do banco traseiro e 13 º C a mais superior ao assento do motorista como mostrado na tabela IV.

A medida de temperatura na zona do painel central excedeu o limite superior do range de medição aceitável (0 a 50 º C), sendo a região de maior temperatura do veículo. Com base na tabela IV, atribuiu-se a incidência maior dos raios solares sobre o vidro do para-brisa como a causa da média elevada de temperatura.

TABELA IV
CARRO SEM PROTETOR – TEMPERATURA E UMIDADE POR POSIÇÃO

Posição	Temperatura Média (º C)	Umidade Relativa (%)
Painel central	58,7	1,5
Assento do motorista	45,3	1,5
Banco Traseiro	42,9	1,6

A. Influência do protetor solar na medição

O protetor solar de para-brisas é um objeto composto de um filme laminado que tem como função refletir os raios incidentes no vidro frontal do carro.

Para verificar a o efeito de uso protetor solar de para-brisas na temperatura e umidade da região do painel do veículo foi desenvolvido um método de medição com o uso do ThingSpeak.

Foram confeccionadas duas estações de medição de UR e temperatura. Uma estação foi posicionada sobre o painel de um veículo sem protetor de para-brisas e a outra foi instalada em um painel com a proteção. Os veículos possuem película nos vidros laterais e traseiros de mesma opacidade (70%). A plataforma online coletou os dados e os plotou em gráficos no mesmo canal. Foram criados 4 campos para o envio de informação para o canal no ThingSpeak. O medidor no carro com proteção enviou os dados para os campos 1 e 2, temperatura e umidade respectivamente, e o segundo medidor enviou os dados para os campos 3 e 4 , temperatura e umidade respectivamente. A aplicação na internet foi configurada para gerar gráficos com a taxa de postagem de 15 segundos, limite mínimo do site. O intervalo da amostragem foi das 12:10 h às 12:35h. Os gráficos gerados são mostrados nas Fig. 42 a 45.

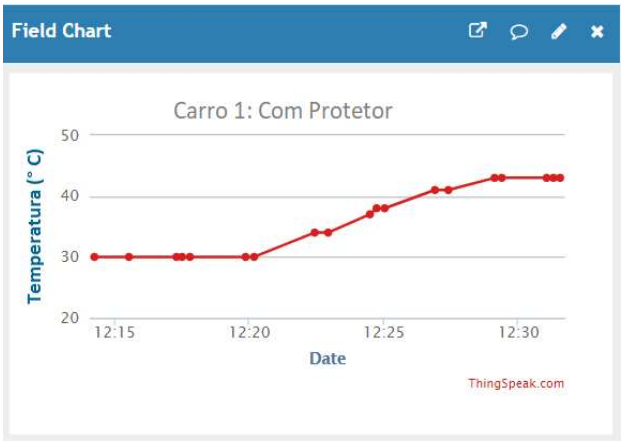

Fig. 42. Temperatura do carro com protetor

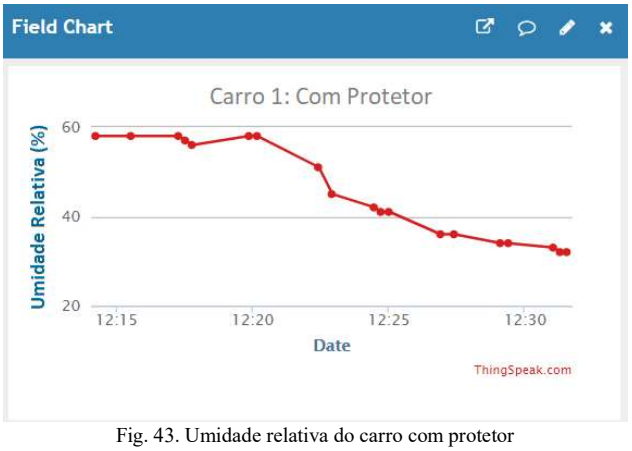

Fig. 43. Umidade relativa do carro com protetor

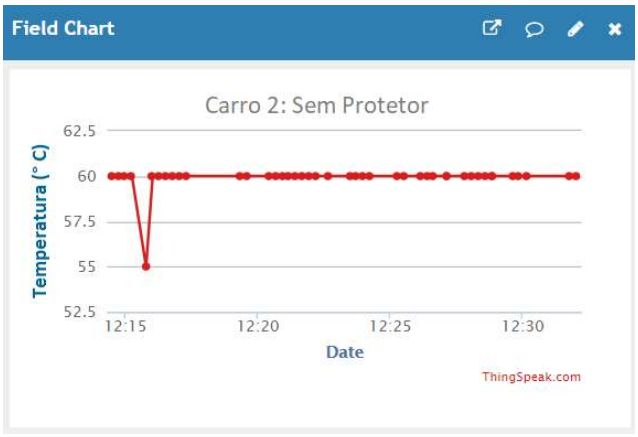

Fig. 44. Temperatura carro sem protetor

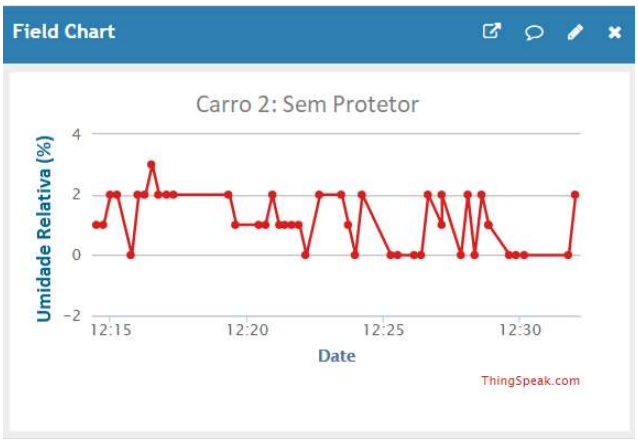

Fig. 45. Umidade relativa do carro sem protetor

Os gráficos do carro com a proteção instalada apresentaram menos pontos plotados e maior espaçamento entre valores seguidos se comparados aos gráficos sem protetor. A barreira física posta com a instalação do protetor reduziu a qualidade do sinal, porém como a taxa de postagem é elevada estes efeitos podem não comprometeram a análise.

Comparando o gráfico de temperatura do carro 1 (Fig. 42) com o gráfico do carro 2 (Fig.44), notou-se, como já esperado uma menor temperatura interna. Com a proteção a grandeza ficou dentro do range, com o valor máximo de 42 °C. Houve uma rampa de elevação seguida de estabilização no valor de pico. Já no carro 2, sem o protetor, a temperatura ficou acima da escala padrão do instrumento e o gráfico dá indícios de saturação.

A umidade do ar no veículo 1 apresentou uma forte queda de 50% nos primeiros 20 min, mas manteve os valores dentro da faixa de medição do instrumento. O valor mínimo foi de 32%. Para o segundo veículo a temperatura ficou abaixo do limite inferior (20%) e os dados plotados não podem ser considerados.

Mesmo com ao anteparo de proteção a exposição direta do ao sol reduz a qualidade do ar do interior do veículo. A média de umidade é classificada como estado de atenção (entre 21 e 30%) para o veículo com a proteção instalada [19]. A temperatura do interior do veículo é elevada e cresce em um curto período. O uso do protetor de para-brisas atenuou os efeitos da exposição solar mesmo no período de incidência elevada.

B. Alerta ao usuário via Twitter

O ambiente interno de veículos é composto de diversos componentes de plástico, couro e metal. A exposição à intensa radiação solar pode danificar estes componentes, modificar a coloração e reduzir a vida útil dos materiais.

O couro, em especial, é um material sensível à temperatura e a umidade do ambiente. Além de fragilizar o material a exposição prolongada ao sol causa uma sensação desagradável ao sentar em bancos revestidos deste material.

Tendo em vista este problema foi desenvolvida uma aplicação para informar ao usuário quando a temperatura da cabine do veículo for superior a 30º C. Este valor foi escolhido para fins de teste desta integração.

O alerta ao usuário foi realizado através da rede social Twitter devido a praticidade ofertada no ThingSpeak com a aplicação REACT. Exemplos de postagens automáticas de alerta ao usuário são mostradas na Fig.46.

Fig. 46. Alertas de temperaturas elevadas ao usuário via Twitter

A integração com o Twitter mostrou-se uma excelente ferramenta de alerta ao usuário. Todas as mensagens foram enviadas conforme a condição era satisfeita.

XII. CONCLUSÃO

Este artigo apresentou um projeto para monitoramento de temperatura e umidade do interior de veículos aplicando conceitos de internet das coisas. O hardware utilizado foi de baixo custo e apresentou algumas limitações de medição de umidade relativa e principalmente de temperatura. O sensor de umidade temperatura DHT11 apresentou desempenho satisfatório para sistemas com protetor de para-brisas instalado. Um resultado relevante obtido nesta pesquisa foi a constatação da influência do protetor de para-brisas na redução da temperatura interna de veículos expostos ao sol diretamente.

O ambiente de prototipagem mostrou-se uma solução para testes rápidos de ligação de componentes sobre bancada, porém para a exposição prolongada o uso da soldagem eletrônica faz-se imprescindível para a qualidade do projeto.

A plataforma de IoT ThingSpeak apresentou uma fácil integração com a estação de medição desenvolvida. A plataforma disponibilizou recursos de análises de dados com o MATLAB, de forma gratuita, em que foram calculadas as médias da temperatura e da umidade do ambiente. O recurso REACT possibilitou a integração com o Twitter como aplicação de alerta ao usuário de temperaturas elevadas no interior do veículo.

Projetos futuros devem ser desenvolvidos para avaliar de maneira mais precisa o impacto na qualidade do ar no interior de veículos expostos ao sol. Uma proposta para estudos posteriores é a relação eficiência de medidores de temperatura e umidade DHT-11, DHT-22 e LM35 com o custo de cada componente.

REFERÊNCIAS

[1] DIAS, Renata Rampin de Freitas. **Internet das coisas sem mistérios:** Uma nova inteligência para os negócios. São Paulo: Netpress Books, 2016. 123 p.

[2] OLIVEIRA, Sérgio de. **Internet das Coisas com ESP8266, Arduino e Raspberry Pi.** São Paulo: Novatec, 2017. 240 p.

[3] CETIC. **TIC Domicílios:** Pesquisa sobre o Uso das Tecnologias de Informação e Comunicação no Brasil. 2016. Disponível em: <http://cetic.br/media/analises/tic_domicilios_2016_coletiva_de_imprensa_2.pdf> Acesso em: 02 abril. 2018.

[4] KODALI, Ravi Kishore; MAHESH, Kopulwar Shishir. A low cost implementation of MQTT using ESP8266. **2016 2nd International Conference On Contemporary Computing And Informatics (IC3I)**,2016. IEEE.

[5] KODALI, Ravi Kishore; MAHESH, Kopulwar Shishir. Low cost ambient monitoring using ESP8266. **2016 2nd International Conference On Contemporary Computing And Informatics (ic3i)**, p.1-4, dez. 2016. IEEE.

[6] JAMES, Jerrin; P, Manu Maheshwar. Plant growth monitoring system, with dynamic user-interface. **2016 Ieee Region 10 Humanitarian Technology Conference (r10-htc)**,p.1-5, dez. 2016. IEEE.

[7] RAY, Partha Pratim. Internet of Things cloud enabled MISSENARD index measurement for indoor occupants. **Measurement**, p.157-165, out. 2016. Elsevier BV.

[8] SAHA, Saraswati; MAJUMDAR, Anupam. Data centre temperature monitoring with ESP8266 based Wireless Sensor Network and cloud based dashboard with real time alert system. **2017 Devices For Integrated Circuit (devic)**,p.1-4, mar. 2017. IEEE.

[9] KUMAR, C. Kishore et al. Internet of things based approach for open precision farming. **2017 International Conference On Advances In Computing, Communications And Informatics (icacci)**, p.1-6, set. 2017. IEEE.

[10] LI, Yangjun; HE, Juan. Design of an intelligent indoor air quality monitoring and purification device. **2017 Ieee 3rd Information Technology And Mechatronics Engineering Conference (itoec)** p.1-4, out. 2017. IEEE.

[11] SWAIN, Kunja Bihari; SANTAMANYU, G.; SENAPATI, Amiya Ranjan. Smart industry pollution monitoring and controlling using LabVIEW based IoT. **2017 Third International Conference On Sensing, Signal Processing And Security (icsss)**, p.1-5, maio 2017. IEEE.

[12] PORTA, Leonardo dalla. **SENSOR DE TEMPERATURA E UMIDADE COM JUMPER – DHT11.** Disponível em: <http://blog.usinainfo.com.br/sensor-de-temperatura-e-umidade-com-jumper-dht11/>. Acesso em: 23 fev. 2018.

[13] IDGNOW. **Em 2021, mercado brasileiro de Internet das Coisas será de US$ 3,29 bilhões.** Disponível em: <http://idgnow.com.br/ti-corporativa/2017/06/18/em-cinco-anos-mercado-de-iot-no-brasil-vai-valer-us-3-29-bilhoes/>. Acesso em: 08 mar. 2018.

[14] COLUMBUS, Louis. **2017 Roundup Of Internet Of Things Forecasts.** 2017. Disponível em: <https://www.forbes.com/sites/louiscolumbus/2017/12/10/2017-roundup-of-internet-of-things-forecasts/#68c6499e1480>. Acesso em: 08 mar. 2018.

[15] MINATEL, Pedro. **ESP8266: O guia básico de Hardware.** 2015. Disponível em: <http://pedrominatel.com.br/pt/esp8266/esp8266-o-guia-basico-de-hardware/>. Acesso em: 06 abr. 2018.

[16] CURVELLO, André. **Apresentando o módulo ESP8266.** 2015. Disponível em: <https://www.embarcados.com.br/modulo-esp8266/>. Acesso em: 05 abr. 2018.

[17] CINTRA, José. **Tutorial:** Programando a NodeMCU (ESP8266) com a IDE do Arduino. 2016. Disponível em:<http://josecintra.com/blog/programando -nodemcu-esp8266-ide arduino/>. Acesso em: 19 mar. 2018.

[18] INEMET. **NORMAIS CLIMATOLÓGICAS DO BRASIL.** Disponível em: <http://www.inmet.gov.br/portal/index.php?r=clima/normaisclimatologicas>. Acesso em: 08 abr. 2018.

[19] CGE. Prefeitura de São Paulo. **Umidade Relativa do Ar.** Disponível em:<https://www.cgesp.org/v3/umidade-relativa-do-ar.jsp>. Acesso em: 08 abr. 2018.

[20] COSSETTI, Melissa Cruz. **Twitter aumenta oficialmente o limite de 140 caracteres para 280.** Disponível em:<https://www.techtudo.com.br /noticias /2017/11/twitter-aumenta-oficialmente-o-limite-de-140-caracteres-para-280.ghtml>. Acesso em: 07 abr. 2018.