Protocolos, Tecnologias, Ferramentas e Laboratórios para Aplicações de Internet das Coisas

Evandro Cantú
Instituto Federal do Paraná
Campus Foz do Iguaçu
evandro.cantu@ifpr.edu.br

Carlos Barros Montez
Departamento de Automação e Sistemas
Universidade Federal de Santa Catarina
carlos.montez@ufsc.br

***Resumo* — A Internet das Coisas permite conectar objetos do mundo físico a Internet, possibilitando a criação de novas e inovadoras aplicações. Neste cenário, são apresentadas neste artigo, primeiramente, uma conceituação e uma proposta de arquitetura simplificada para aplicações de Internet das Coisas. Em seguida, são apontadas as principais tecnologias e protocolos de comunicação que permitem suportar estes tipos de aplicações. Por fim, são descritos um conjunto de ferramentas de software e dispositivos de hardware, que podemos utilizar para a construção de protótipos e realizar provas de conceito para aplicações de Internet das Coisas. O uso destas tecnologias e ferramentas é ilustrado através de laboratórios e exemplos de aplicação, com destaque para a ferramenta de programação Node-RED. Desta forma, pretendemos divulgar neste artigo informações relevantes sobre Internet das Coisas, tema este que está em alta tanto no mercado quanto na comunidade acadêmica.**

***Palavras-chave* — Internet das Coisas; Node-RED; Tecnologias para Internet das Coisas.**

Abstract* — *The Internet of Things allows connecting objects from the physical world to the Internet, enabling the creation of new and innovative applications. In this scenario, this article presents, first, a conceptualization and a simplified architecture proposal for Internet of Things applications. Then, the main communication technologies and protocols that support these types of applications are pointed out. Finally, a set of software tools and hardware devices are described, which we can use to build prototypes and perform proof of concept for Internet of Things applications. The use of these technologies and tools is illustrated through laboratories and application examples, with emphasis on the programming tool Node-RED. In this way, we intend to disseminate with this article relevant information about Internet of Things, a topic that is on the rise both in the market and in the academic community.

I. INTRODUÇÃO

A Internet das Coisas pode ser vista como um extensão da Internet, na qual objetos do mundo físico são conectados a rede, permitindo a construção de novas aplicações, as quais estão mudando nosso mundo. Suas áreas de aplicação são ilimitadas, incluindo bens e consumo, como *smartphones*, televisores, geladeiras e outros aparelhos inteligentes. Aplicações pessoais voltadas ao esporte e saúde, apoiadas por dispositivos portáteis ou vestíveis, visando monitorar parâmetros de saúde ou de atividades físicas. Aplicações para residências inteligentes, visando automatizar e monitorar a iluminação, a climatização, a segurança ou o consumo de energia em uma residência. Aplicações para cidades inteligentes, como semáforos automatizados, controle da iluminação pública, controle e monitoramento do transporte coletivo e diversas outras possibilidades. Na área comercial ou de logística, aplicações de *marketing*, controle de estoque e rastreamento de produtos [1].

Outras duas importantes áreas de aplicação para a Internet das Coisas são na indústria e na agricultura. Na indústria, a Internet das Coisas tem sido apontada como principal elemento da chamada Indústria 4.0. Este termo se refere à quarta revolução industrial, impulsionada pela tecnologias da Internet para criar produtos, linhas de produção e serviços inteligentes. Neste cenário, além dos objetos inteligentes e do processamento na nuvem da Internet, as tecnologias de comunicação têm papel fundamental para a troca de informações entre os diferentes tipos de sensores, controladores e atuadores [2].

Na agricultura, técnicas computacionais, associadas aos dispositivos inteligentes da Internet das Coisas, são responsáveis pela quarta revolução agrícola. A título de curiosidade, primeira revolução agrícola, acompanhando a revolução industrial, foi o desenvolvimento do arado mecanizado e o uso de tratores. A segunda, a revolução química e o uso de fertilizantes. A terceira, a engenharia genética, permitindo cultivo em múltiplas estações do ano [3].

Como no caso das aplicações industriais, na agricultura as redes de comunicação, em particular as redes sem fio de longa distância, têm papel fundamental na troca de informações entre sensores e atuadores localizados em campo.

Neste cenário amplo de aplicações para a Internet das Coisas, pretendemos neste artigo, primeiramente, conceituar brevemente e apresentar uma arquitetura simplificada para as aplicações de Internet das Coisas. Em segundo, apontar as principais tecnologias e protocolos de comunicação, disponíveis atualmente, para o suporte das aplicações de Internet das Coisas. Por fim, apresentar um conjunto de ferramentas e dispositivos, incluindo componentes de software e hardware, que podemos utilizar para construção de protótipos e provas de conceito para aplicações de Internet das Coisas. O uso destas tecnologias e ferramentas é ilustrado através de laboratórios e exemplos de aplicação.

II. Conceitos e Arquitetura para Internet das Coisas

A. Conceitos sobre Internet das Coisas

A ideia principal da Internet das Coisas, ou em inglês *Internet of Things* (IoT), é conectar "coisas" (sensores, atuadores, dispositivos, máquinas, pessoas etc) e realizar o armazenamento e processamento de dados na nuvem da Internet para fins de monitoramento e controle. Portanto, em sua definição mais simples, a Internet das Coisas pode ser considerada como a intersecção de coisas, dados e a Internet [4].

Os requisitos fundamentais para monitorar coisas em qualquer lugar do mundo são: identificação única para as coisas; habilidade de sensoriamento (ou atuação) de alguma informação sobre as coisas; habilidade de comunicação entre as coisas e a Internet e habilidade de controle e gerenciamento das coisas [4].

B. Uma Arquitetura para Internet das Coisas

Os requisitos descritos na seção anterior sugerem uma proposta de arquitetura em quatro camadas para a Internet das Coisas, ilustrada na Figura 1 (adaptado de [4]).

Na primeira camada desta arquitetura estariam os dispositivos sensores e atuadores transmitindo (ou recebendo) pequenas mensagens de dados diretamente ou até *gateways* conectados a Internet.

A segunda camada seria a camada de conectividade em rede para Internet das Coisas, transportando dados entre os dispositivos e a nuvem da Internet.

A terceira camada seria uma plataforma de serviços para Internet das Coisas, comumente hospedada na nuvem da Internet, conectando as coisas com as aplicações e provendo mecanismos e ferramentas para gerenciamento e tratamento das informações. Esta plataforma pode também utilizar dados de outras fontes ou sistemas para complementar as informações coletadas pelos sensores.

A quarta camada seria a camada de aplicação, dedicada a uma aplicação específica, para operar e fornecer a interface usuário a um sistema de Internet das Coisas.

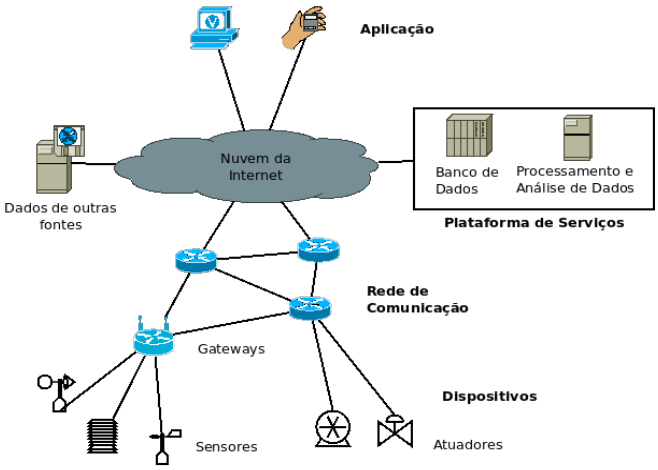

Fig. 1. Proposta de uma arquitetura para Internet das Coisas.

III. Protocolos de Comunicação para Internet das Coisas

As aplicações Web que conhecemos são suportadas pela arquitetura Internet com quatro camadas de protocolos [5]. A camada de enlace/física, responsável pela comunicação entre computadores diretamente conectados através de um enlace. A camada de enlace/física, neste texto considerada como uma única camada, normalmente é implementada numa interface de rede, como nas redes locais *Ethernet* ou nas redes locais sem fio. A camada de rede cujo principal componente é o protocolo IP (*Internet Protocol*). A camada de transporte com os protocolos TCP (*Transport Control Protocol*) e UDP (*User Datagram Protocol*). E a camada de aplicação, cada qual com seu protocolo específico, como por exemplo o protocolo HTTP (*Hypertext Transfer Protocol*) da aplicação Web.

Entretanto, na Internet das Coisas muitos dispositivos sensores e atuadores são restritos em termos de processamento, memória e consumo de energia. Estas restrições limitam a capacidade de implementar a pilha completa dos protocolos da Internet em cada dispositivo [6].

Além das restrições em termos de hardware dos dispositivos, o modelo de comunicação cliente/servidor, comumente utilizado na Web, não é apropriado para as aplicações de Internet das Coisas. Na Internet das Coisas os dados fluem em um sentido, dos sensores para a Internet, ou em sentido contrário no caso de atuadores. Além disto, podemos ter múltiplos sensores e também múltiplas aplicações com interesse na informação dos sensores. Portanto, o modelo de comunicação mais apropriado para este tipo de aplicação é o publicador/subscritor [4, 6, 7]. Neste modelo, quando um sensor tem uma informação disponível ele a publica como um tópico em um servidor intermediário, chamado *broker*. Por outro lado, as aplicações podem subscrever tópicos de interesse, recebendo a informação de um sensor quando ela for publicada.

Na sequência desta seção, descreveremos, de forma sucinta, os principais protocolos e tecnologias de comunicação, disponíveis atualmente, que podem ser utilizados para implementar aplicações de Internet das Coisas.

A. Camada de Enlace para sensores e atuadores

Na Internet das Coisas, os dispositivos sensores e atuadores geralmente são restritos em termos de processamento, memória e consumo de energia. Portanto, dependendo da aplicação e do ambiente onde ficará localizado o sensor ou atuador, diferentes tecnologias de enlace podem ser utilizadas.

Para os ambientes industriais as redes IEEE 802.15.4[1] [6, 7] apresentam soluções de comunicação sem fio com baixa taxa de transmissão e baixo consumo de energia, com destaque para a tecnologia *Zigbee*[2].

Para sensores localizados em ambientes externos, como nas aplicações de agricultura [3], as redes LoRaWAN (*Long*

1 **IEEE 802**: O IEEE (*Institute of Electrical and Electronics Engineers*) padroniza as redes locais de computadores, reunidas no padrão IEEE 802, como por exemplo, as redes Ethernet (IEEE 802.3) e as redes WiFi (IEEE 802.11).

2 **Zigbee Alliance**: Standard-bearer of the open IoT. https://zigbeealliance.org/.

Range – Wide Area Networks) [6, 7] oferecem comunicação sem fio de baixa potência para longas distâncias.

Para dispositivos localizados em ambientes internos, tecnologias conhecidas como WiFi, *Bluetooth*, *Bluetooth Low Energy* (BLE) ou mesmo comunicação serial podem ser utilizadas.

B. Camada Rede para Internet das Coisas

Um dos requisitos fundamentais para monitorar coisas é termos uma identificação única para cada dispositivo. Muitas aplicações de Internet das Coisas utilizam identificação de objetos usando RFID (*Radio Frequency Identification*) [6], a qual permite identificar objetos a partir de informações gravadas em uma *tag*. A *tag* é formada por um microchip com informações gravadas e uma antena para receber e transmitir sinal. O leitor lê as informações gravadas na *tag* por meio de um rádio transmissor e receptor e passa para um programa para análise do código RFID.

Entretanto, caso seja necessário conectar cada dispositivo à Internet será necessário um endereço IP para cada dispositivo. Assim, surge a necessidade de se utilizar a última versão do protocolo IP, o IPv6, o qual possui endereçamento suficiente para os bilhões, ou mais, de dispositivos de Internet das Coisas [6, 7].

Por outro lado, caso o dispositivo seja limitado para implementar a pilha completa de protocolos da Internet, o mesmo pode enviar seus dados a um *gateway* intermediário, que por sua vez encaminha as informações a nuvem da Internet. Neste caso, os dispositivos são identificados por seus respectivos *gateways* na Internet.

Outras tecnologias de rede que podem ser utilizadas para conectar *gateways* remotos a Internet são as redes de telefonia móvel, como 4G e 5G.

Para os dispositivos que utilizem as redes sem fio IEEE 802.15.4 e precisem se comunicar através da Internet, é necessário adaptar o pacote da camada de rede para que o mesmo possa ser transportado por um quadro da camada de enlace. Para tanto, foi desenvolvido o padrão 6LowPAN [6, 7] que adapta o IPv6 para as limitações de tamanho do quadro impostas pelas redes IEEE 802.15.4.

C. Camada de Transporte para Internet das Coisas

A camada de transporte da Internet é baseada nos protocolos TCP e UDP. O TCP é o principal protocolo utilizado na Web. É um protocolo orientado a conexão, que utiliza reconhecimentos e retransmissões para oferecer um serviço de transmissão garantida de informações. Estas características do TCP não são adequadas para a comunicação entre os dispositivos da Internet das Coisas [6]. Por exemplo, para um sensor que envia periodicamente dados de uma grandeza medida, a perda de um dado pode não ser crítica, uma vez que a informação é redundante em função do número de medidas realizadas.

Portanto, na Internet das Coisas, a comunicação entre dispositivos pode ser mais eficiente com o uso do protocolo UDP. Uma mensagem transmitida pelo protocolo UDP é simplesmente enviada. Não há reconhecimentos ou retransmissões. Esta característica é conhecida como melhor esforço (*best effort*).

D. Camada Aplicação para Internet das Coisas

Como comentado na introdução desta seção, o modelo cliente/servidor comumente utilizado na Internet não é apropriado para as aplicações de Internet das Coisas.

Para aplicações de Internet das Coisas o modelo de comunicação mais apropriado é o publicador/subscritor, como utilizado no protocolo MQTT (*Message Queue Telemetry Transport*) [4, 6, 7], o qual é descrito na subseção 1.

Por outro lado, caso seja necessária uma comunicação direta com um dispositivo de Internet das Coisas utilizando o modelo cliente/servidor, um protocolo alternativo é o CoAP (*Constrained Application Protocol*) [6, 7], descrito na subseção 2.

1) Protocolo MQTT

O protocolo MQTT [8, 9] é baseado no princípio de publicar mensagens (*publisher*) e subscrever tópicos (*subscriber*) para receber mensagens de interesse.

Um cliente pode subscrever tópicos e receber mensagens de atualizações quando algo for publicado sobre este tópico. Alternativamente, um cliente pode publicar mensagens relativas a um dado tópico e deixá-la disponível para os assinantes. Neste modelo de comunicação o publicador e o subscritor não se comunicam diretamente. A comunicação é realizada por um terceiro componente, o *broker*.

As subscrições podem ser explícitas ou organizadas hierarquicamente, com o cliente usando caracteres coringa (# ou +) para receber mensagens de uma variedade de tópicos.

A Figura 2 ilustra um cenário com três sensores publicando mensagens e dois subscritores recebendo dados publicados.

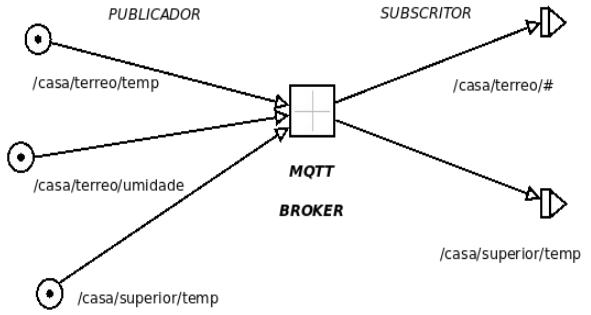

Fig. 2. Modelo publicador/subscritor e uso de caracteres coringa.

O MQTT define três níveis de qualidade de serviço para a troca de mensagens:

(0) *best effort*, envia a mensagem uma vez;

(1) envia a mensagem uma ou mais vezes, podendo haver mensagens duplicadas;

(2) assegura o recebimento de apenas uma mensagem.

Outra característica do protocolo MQTT é a possibilidade de reter mensagens publicadas no *broker* e tê-las disponível, imediatamente, para um novo subscritor deste tópico.

Também é possível no MQTT termos sessões persistentes, as quais permitem a um cliente subscritor manter suas assinaturas mesmo após uma desconexão. Neste

caso, as subscrições do cliente continuaram ativas após a próxima conexão.

Outra característica interessante do MQTT é chamada testamento, na qual quando um cliente se conecta a um servidor e pode informar que possui mensagens importantes que deveriam ser publicadas em caso de desconexão involuntária. Esta característica é importante nas aplicações de Internet das Coisas para avisar os assinantes, por exemplo, quando um sensor perdeu a conexão com a rede.

Outros detalhes do protocolo MQTT, assim como um estudo de suas trocas de mensagens usando Wireshark[3], podem consultados em [10].

2) *Protocolo CoAP*

O CoAP [7] é uma alternativa mais leve ao HTTP, com alvo nos dispositivos limitados em termos de energia e comunicação. O CoAP usa UDP, ao invés do TCP usado pelo HTTP, reduzindo o *overhead* de mensagens ocasionado pela abertura e encerramento, reconhecimento e retransmissões de uma conexão TCP.

As mensagens CoAP seguem a arquitetura REST (*Representational State Transfer*) para criar, ler, atualizar ou apagar de dados de aplicação em servidores remotos, através dos métodos GET, PUT, POST e DELETE de modo similar ao HTTP.

O CoAP usa mensagens curtas com cabeçalho de tamanho fixo que podem ser seguidas por opções e dados. A entrega das mensagens pode ser confirmável ou não confirmável. No caso de mensagens confirmáveis, o CoAP utiliza temporizações e retransmissões para garantir a entrega das mensagens.

Por utilizar a arquitetura REST, o CoAP pode ser integrado ao HTTP por meio de um *proxy*, permitindo, por exemplo, que um usuário utilizando a Web e o HTTP interaja com dispositivos restritos que suportem o CoAP.

IV. Tecnologias e Ferramentas para Internet das Coisas

Nesta seção são apresentadas algumas ferramentas de software e dispositivos de hardware que podemos utilizar para implementar protótipos ou realizar testes de conceitos de aplicações para Internet das Coisas. O uso das ferramentas está exemplificado na forma de laboratórios na seção V.

A principal ferramenta utilizada nos laboratórios foi o ambiente de programação Node-RED [11], o qual permite interligar dispositivos físicos, funções para tratamento de dados e serviços em nuvem.

Outras ferramentas de software livre utilizadas foram o *broker* MQTT Mosquitto [12] e os contêineres Docker [13], estes últimos utilizados para implementar instâncias de software de forma independente do sistema operacional hospedeiro.

Como dispositivos de hardware, foram utilizados a plataforma Arduíno, os microcontroladores ESP8266 e ESP32 e o computador de placa única Raspberry Pi, todos facilmente disponíveis no mercado.

3 **Wireshark**: Programa para captura de pacotes e análise de protocolos. https://www.wireshark.org/.

A. *Ambiente de programação Node-RED*

O Node-RED [11] é um ambiente de programação *low code* no qual blocos são interligados na forma de fluxos, utilizando uma interface de navegador, como mostrada na Figura 3.

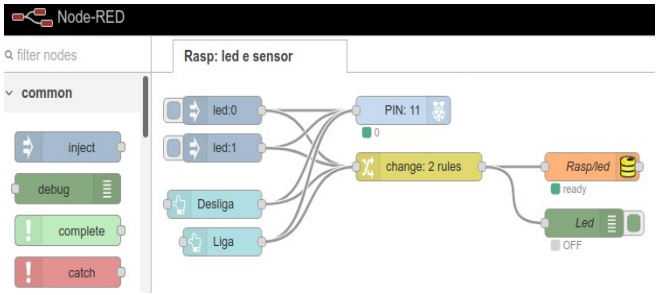

Fig. 3. Interface de programação do Node-RED .

O Node-RED possui diversos blocos funcionais, os quais podem ser arrastados formando fluxos de informações e criando aplicações.

Os blocos estão agrupados por funcionalidades, como entradas e saídas de dados, funções para tratamento e manipulação de dados, blocos para comunicação em rede (como MQTT, HTTP, TCP, UDP, comunicação serial etc), blocos para realizar o controle de fluxo e outros.

Dependendo da aplicação, outros blocos podem ser instalados, como por exemplo, blocos para comunicação com o Arduíno, blocos comunicação para protocolo CoAP, blocos especiais para tratamento matemático de dados e diversas outras possibilidades.

O Node-RED também permite a interação com outros ambientes e sistemas em nuvem, como a plataforma digital Firebase[4] da Google e a plataforma AWS[5] da Amazon, as quais oferecem acesso a banco de dados e auxiliam no desenvolvimento de aplicações Web e para dispositivos móveis.

O Node-RED também permite a construção de painéis de controle, ou *dashboards*, para facilitar a visualização dos dados e o controle das aplicações.

O Node-RED pode ser instalado localmente em uma máquina, assim como, também pode ser instalado como uma instância em um contêiner Docker, ficando independente do sistema operacional hospedeiro, o que também facilita uma possível migração para outro ambiente.

Outra alternativa, para ter acesso via Web da aplicação através de endereço IP público, é instalar o Node-RED em um ambiente em nuvem, como em um console da AWS ou num contêiner na plataforma em nuvem da Digital Ocean[6].

B. *broker Mosquitto*

O Mosquitto [12] é um *broker* MQTT *open source*, que pode ser utilizado desde em computadores de placa única até em servidores. O Mosquitto implementa o modelo publicador/subscritor e pode ser utilizado em aplicações de Internet das Coisas, as quais fazem uso de sensores, atuadores, microcontroladores e outros dispositivos programáveis.

4 **Firebase Google**. https://firebase.google.com.
5 **AWS Amazon**. https://aws.amazon.com.
6 **Digital Ocean**. https://www.digitalocean.com/.

O Mosquitto também oferece comandos de linha para serem utilizados em um terminal Linux, como o `mosquitto_pub` e `mosquitto_sub` para publicar e subscrever no *broker*, respectivamente. Estes comandos são bastante úteis para testar a comunicação com sensores e atuadores que se comunicam usando o protocolo MQTT.

O Mosquitto também disponibiliza bibliotecas para implementação de clientes MQTT, como a biblioteca `PubSubClient.h` [14] desenvolvida para os microcontroladores Arduíno, ESP8266 e ESP32.

O Mosquitto também pode ser instalado localmente em uma máquina ou como uma instância em um contêiner Docker.

C. Contêiner Docker

O Docker [13] é um contêiner, que é uma unidade padronizada de software que permite aos desenvolvedores isolar suas aplicações do meio no qual vai rodar.

Os contêineres são tecnologias de virtualização operando no nível do sistema operacional, incluindo somente a parte do sistema operacional e bibliotecas necessárias para rodar a aplicação desejada.

Como afirma [4], os contêineres são fundamentais para implementar instâncias em sistemas de para computação em nuvem (*cloud computing*) ou para computação em névoa (*fog computing*). Estas duas modalidades de computação são fundamentais nos sistemas de Internet das Coisas, permitindo um desacoplamento espacial entre componentes físicos do sistema e das funções de gerenciamento e controle.

Num sistema de Internet das coisas, a computação em névoa ocorre quando o tratamento de dados acontece próximo dos sensores ou atuadores, por exemplo, em um *gateway* responsável por posteriormente enviar dados a nuvem da Internet.

Os conceitos básicos que caracterizam o Docker são: a imagem, que define o sistema operacional base, pacotes, bibliotecas e tudo aquilo que for necessário para rodar uma aplicação; o contêiner, que é a instância criada de uma imagem rodando com comando apropriado; e o *dockerhub*, que é o repositório de onde podemos baixar ou subir imagem.

Quando se lança um comando para instalar uma aplicação no Docker, ele primeiro verifica se a imagem já está disponível, senão busca a imagem no *dockerhub* e instala todas as dependências e requisitos de software necessário para a aplicação. Uma vez rodando, a instância da aplicação se comporta como se estivesse rodando localmente na máquina.

Como comentado anteriormente, tanto instâncias do Node-RED quanto do Mosquitto podem ser instalados no Docker. Além disto, sistemas operacionais Linux dedicados a dispositivos de hardware restrito em termos de processamento e memória, como o Alpine Linux[7] ou Contiki[8], podem ser instalados e testados facilmente no Docker.

7 **Alpine Linux**: Small, Simple, Secure. https://alpinelinux.org/.

8 **Contiki**: OS for Next Generation IoT Devices. https://www.contiki-ng.org/.

D. Hardware para Internet das Coisas

Em termos de dispositivos de hardware para monitorar e controlar sensores ou atuadores temos boas alternativas disponíveis no mercado, como as plataformas baseadas no Arduíno ou no ESP8266 e ESP32, assim como o computador de placa única Raspberry Pi. Todos estes dispositivos foram testados e experimentados nos laboratórios descritos na seção V.

O mercado também oferece toda sorte de sensores e atuadores que podem ser utilizados em aplicações de Internet das Coisas, os quais podem ser facilmente monitorados ou controlados por estes dispositivos programáveis. Encontramos desde sensores analógicos simples, que oferecem como saída uma tensão variável que pode ser lida por uma entrada analógica do microcontrolador, até sensores digitais sofisticados que enviam a informação de forma serial ao microcontrolador. Neste último caso, usam-se bibliotecas apropriadas, disponibilizadas pelo fabricante, que facilitam a programação e a obtenção das grandezas medidas.

1) Plataforma Arduíno

A plataforma de hardware livre Arduíno[9] é uma interessante e didática opção para construção de protótipos para Internet das Coisas. Pesam a favor do Arduíno, a facilidade de uso, o grande número de exemplos prontos e documentados [15] disponíveis através de sua interface de programação (IDE), a grande variedade de placas de expansão (*shields*) e a grande quantidade de aplicações e exemplos disponíveis na Internet.

O Arduíno possui vários modelos de placas, com diferentes funcionalidades, todas dispondo de entradas e saídas digitais, entradas analógicas, saídas PWM permitindo realizar controles analógicos, interfaces seriais e outras características. Placas de expansão, como por exemplo um *shield* Ethernet, permitem que o Arduíno se comunique diretamente com a Internet.

2) Plataformas ESP8266 e ESP32

O ESP8266 é um microcontrolador que possui uma interface *WiFi* e vem sendo muito utilizado em aplicações de Internet das Coisas.

Uma versão moderna é o ESP32, que possui possui processador Dual Core, interface *WiFi*, *Bluetooth* e vários sensores embutidos.

Estes dois microcontroladores podem ser encontrados em plataformas de desenvolvimento que facilitam a prototipagem. Neste caso, vem montados em placas com fonte de alimentação embutida e conectores para acesso aos pinos de entrada e saída do microcontrolador facilitando o uso com matrizes de contato. Para programação destes microcontroladores pode-se utilizar a IDE do Arduíno, também com diversos exemplos prontos [16], incluindo exemplos para comunicação *WiFi*, *Bluetooth, Bluetooth Low Energy* (BLE) e outros.

Outra versão interessante é o ESP32 WiFi LoRa, que inclui interface para redes LoRaWAN, voltadas a transmissão de dados com baixa taxa de transmissão, baixo consumo de energia e longas distâncias. Esta é uma boa opção para sensores remotos localizados em ambientes externos, como nas aplicações em agricultura.

9 **Arduino**. https://www.arduino.cc/.

3) Raspberry Pi

O Raspberry Pi[10] é um computador de placa única com processador poderoso, memória RAM de até 8 Gbytes, interfaces Ethernet e Wifi, portas USB, interfaces HDMI e dezenas de pinos de entrada e saída digitais.

O Raspberry Pi possui um sistema operacional específico, o Raspbian, baseado no Linux, que deve ser instalado em um cartão de memória. A instalação e a atualização de programas é realizada da mesma forma que qualquer sistema operacional Linux. Além disto, é possível acessá-lo remotamente via terminal utilizando SSH (*Secure Shell*) ou com um servidor VNC (*Virtual Network Computing*) que permite a visualização da interface gráfica do Raspberry Pi através de uma conexão remota segura.

As entradas e saída digitais do Raspberry Pi, ou pinos GPIO (*General Purpose Input/Output*), podem ser utilizados para monitoramento e controle de sensores e atuadores. Estes pinos podem ser programados com *scripts* ou com a linguagem de programação Pyton utilizando bibliotecas apropriadas.

O Raspberry Pi não possui entradas analógicas. Caso seja necessário monitorar um sensor analógico é necessário acoplar um módulo externo ADC (*Analog to Digital Convertor*).

Para aplicações de Internet das Coisas, uma possibilidade interessante é utilizar o Node-RED rodando diretamente no Raspberry Pi. Neste caso, o acesso aos pinos GPIO é realizado diretamente por blocos de entrada e saída disponibilizados no Node-RED.

Outra aplicação interessante para o Raspberry Pi em aplicações de Internet das Coisas é utilizá-lo como *gateway*, recebendo e tratando dados recebidos de outros dispositivos antes de enviá-los a nuvem da Internet. Além disto, ele também pode ser utilizado para computação em névoa, quando for necessário processamento de dados próximo dos dispositivos de Internet das Coisas.

V. Laboratórios com Exemplos de Aplicação para Internet das Coisas

Nesta seção são apresentados alguns laboratórios com exemplos de aplicação para Internet das Coisas, utilizando as ferramentas de software e dispositivos de hardware descritos na seção anterior.

A. Laboratório: Node-RED controlando Arduíno com protocolo Firmata

Neste exemplo foi utilizado um Arduíno UNO controlado pelo Node-RED com o protocolo Firmata. Para isto foi instalado no Node-RED módulo `node-red-node-arduino`.

A comunicação é realizada com os blocos do Node-RED `arduino-in` e `arduino-out` que recebem e enviam comandos ao Arduíno pela serial USB.

No Arduíno e deve ser carregado o *sketch* (forma como são chamados os programas no Arduíno) `StandardFirmata`, disponível nos exemplos da IDE do Arduíno.

A Figura 4 ilustra os blocos (blocos na cor azul) `arduino-out`, que recebe valores de entrada *true* ou *false* para comandar o led conectado ao pino 13 do Arduíno, e o bloco `arduino-in`, para monitorar o estado do pino digital 2, cujo estado é mostrado na janela *debug* da direita.

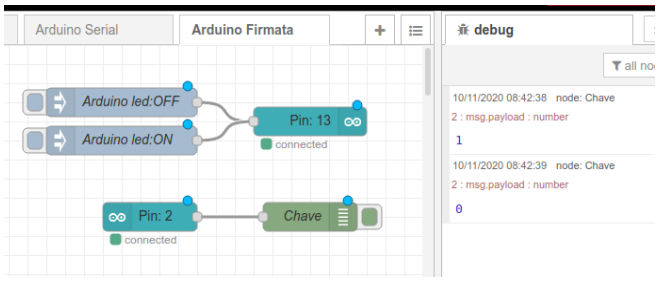

Fig. 4. Interação do Node-RED com Arduíno com Firmata.

B. Laboratório: Interação entre Node-RED e Arduíno usando comunicação serial

Para interação com Arduíno usando comunicação serial deve ser instalado no Node-RED o módulo `node-red-node-serialport`.

Neste exemplo foi utilizado um Arduino UNO**,** ao qual foram acoplados um led para controle e um sensor de temperatura e umidade DHT22. O *sketch* no Arduíno envia periodicamente os dados de temperatura e umidade pela porta serial e espera comandos para o led[11]. Os valores de temperatura e umidade são enviados na forma de `strings` de caracteres separados por vírgula. A Figura 5 ilustra a comunicação serial com o Arduíno. O bloco `serial-in` recebe valores de temperatura e umidade e são mostrados na janela *debug* da direita e o bloco `serial-out` envia comandos para o led (blocos na cor marrom).

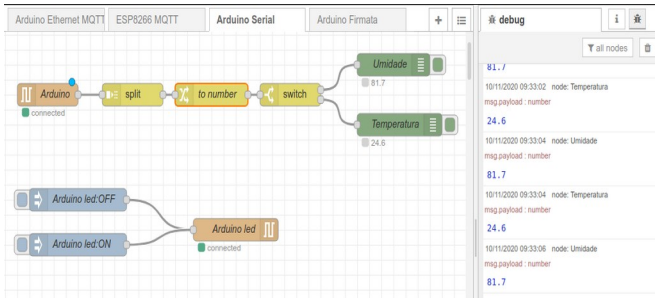

Fig. 5. Interação do Node-RED com Arduíno com comunicação serial.

Os dados de umidade e temperatura recebidos pela entrada serial são separados na forma de vetor pelo nó `split`, convertidos para números pelo nó to `number` e separados em duas saídas pelo nó `switch` (blocos na cor amarelo queimado).

C. Laboratório: Interação do Node-RED com Arduíno via broker Mosquitto

Para interação com Arduíno via *broker* MQTT deve ser instalado no Node-RED módulo `node-red-contrib-mqtt-broker`.

Neste experimento foi utilizado um Arduino UNO com um *Shield* Ethernet, rodando a biblioteca `PubSubClient.h`

10 **Raspberry Py**. https://www.raspberrypi.org/.

11 Detalhes do código fonte utilizado no Arduíno podem ser obtidos em: http://wiki.foz.ifpr.edu.br/wiki/index.php/Node-RED.

[14] para interagir com um *broker* Mosquitto e clientes publicadores e subscritores através do protocolo MQTT.

O *broker* Mosquitto pode ser instalado em qualquer máquina acessível pela Internet. No exemplo foi instalado em uma máquina da rede local onde estava rodando o Node-RED.

O *sketch* rodando no Arduino conecta ao *broker* Mosquitto e informa um testamento para o tópico `arduino/status` com mensagem `off-line`. Esta mensagem será publicada caso o Arduino seja desconectado involuntariamente da rede. Em seguida publica para o tópico `arduino/status` a mensagem `on-line` anunciando que está ativo. O Arduíno ainda subscreve o tópico `arduino/led` para receber comandos para um led. Caso receba mensagem para este tópico, aciona o led conforme o comando recebido[12].

A Figura 6 ilustra o bloco `mqtt-out`, que publica no *broker* o controle para o led do Arduíno no tópico `arduino/led`, e o bloco `mqtt-in`, que subscreve o tópico `arduino/status` (blocos na cor roxa).

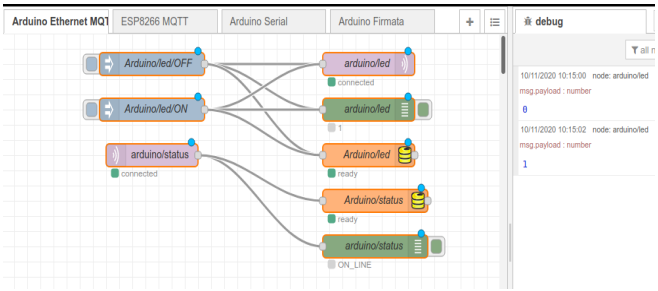

Fig. 6. Interação do Node-RED com Arduíno com MQTT.

Neste exemplo também é ilustrado uma interação com um banco de dado em nuvem no Firebase da Google (blocos na cor laranja). Para tal foi instalado no Node-RED o módulo `node-red-contrib-firebase`.

A Figura 7 ilustra o banco de dados no Firebase, o qual também inclui dados de dos laboratórios descritos nas subseções D e F.

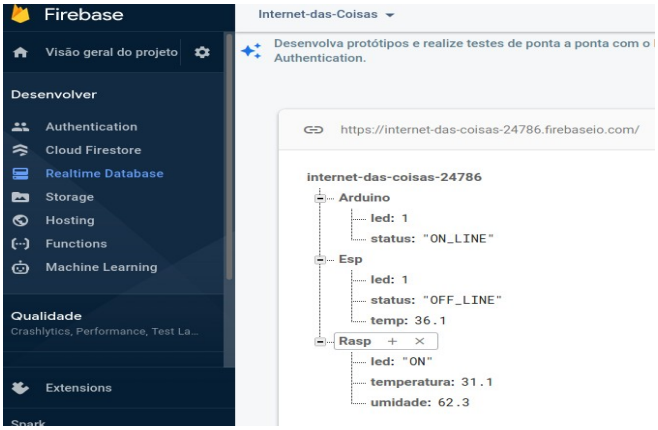

Fig. 7. Banco de dados Firebase da Google.

12 Detalhes do código fonte utilizado no Arduíno, ou no ESP8266 do laboratório descrito na subseção D, podem ser obtidos em: http://wiki.-foz.ifpr.edu.br/wiki/index.php/MQTT_e_Arduino_ou_ESP.

D. *Laboratório: Interação do Node-RED com ESP8266 via broker Mosquitto*

Exemplo similar ao anterior utilizando o ESP8266, ao qual também foi adicionado um sensor de temperatura LM35.

No caso do ESP8266 a conexão com a rede foi realizada através de interface Wifi e também foi utilizada a biblioteca `PubSubClient.h` [14].

A Figura 8 ilustra o cenário deste exemplo, o qual também salva valores no banco de dados do Firebase.

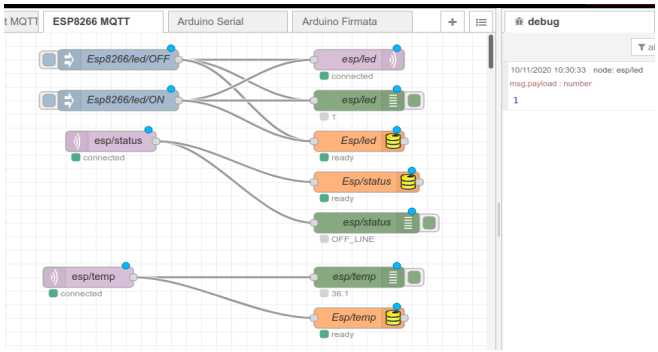

Fig. 8. Interação do Node-RED com ESP8266 com MQTT.

E. *Laboratório: Interação com Arduíno via protocolo CoAP*

Este exemplo ilustra uma interação com um Arduíno utilizando o protocolo CoAP. Foi utilizado um Arduino UNO com um *Shield* Ethernet e a biblioteca `CoAP-simple-library` [17]. Para testes foi utilizado um *sketch* baseado no exemplo `coapserver`, que acompanha a biblioteca.

Neste exemplo, o Arduíno envia periodicamente uma requisição (GET) para um servidor CoAP solicitando a hora relógio e imprime no monitor serial. Além disto, aceita requisições de um cliente (PUT), permitindo comandar um led conectado a uma de suas saídas digitais.

Como servidor CoAP, foi utilizada uma máquina Linux rodando o servidor `coap-server`, baseado na biblioteca `libcoap`. Esta biblioteca também disponibiliza um cliente, o `coap-client`, o qual pode ser utilizado para testes com a execução de comandos a partir de um terminal Linux.

Também como cliente, foi utilizado o Node-RED com o módulo `node-red-contrib-coap` e o bloco `coap-request`, como ilustra a Figura 9 (bloco na cor azul).

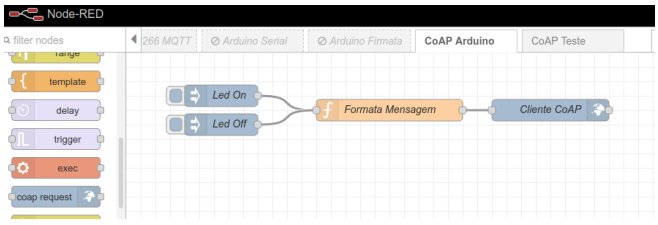

Fig. 9. Interação com Arduíno via CoAP.

F. *Laboratório: Node-RED rodando em Raspberry Pi*

Com um Raspberry Pi rodando o Node-RED é fácil de interagir com sensores e atuadores conectados as portas GPIO através da Web.

Neste laboratório foi utilizado um Raspberry Pi 4, no qual foi instalado o Node-RED e conectados a pinos GPIO um sensor de temperatura e umidade DHT22 e um led para controle.

Para leitura do DHT22 foi instalado no Node-RED módulo `node-red-contrib-dht-sensor`.

A Figura 10 ilustra a parte do exemplo com o controle do led.

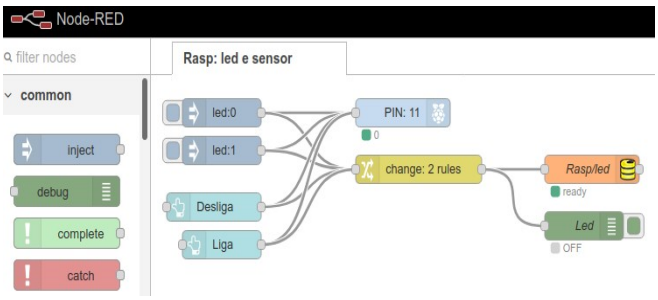

Fig. 10. Node-RED rodando no Raspberry Pi.

A Figura 11 ilustra a leitura do sensor (bloco azul claro). Algumas funções adicionais foram necessárias para separar os dados fornecidos pelo sensor DHT22 em temperatura e umidade, os quais também foram salvos no Firebase (blocos laranja).

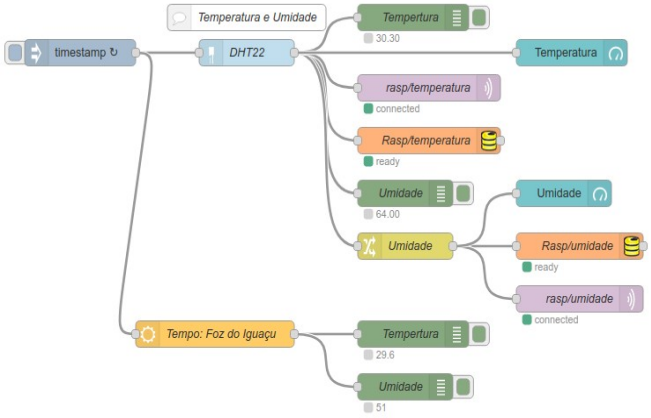

Fig. 11. Node-RED rodando no Raspberry Pi.

Outro destaque do laboratório foi a utilização de módulo para acesso a plataforma *Open Weather*[13] para comparar os valores medidos pelo sensor com dados públicos fornecidos por esta plataforma. Para isto foi instalado no Node-RED módulo `node-red-contrib-weather` (bloco na cor laranja claro).

O Node-RED também permite construir painéis de controle, ou *dashboards*, para monitorar e controlar aplicações de Internet das Coisas. Para isto, deve ser instalado o módulo `node-red-dashboard`. Nas Figuras 10 e 11 os blocos azuis mais claros implementam os controles do *dashboard*.

13 **OpenWeather** global services. https://openweathermap.org/.

A Figura 12 ilustra uma *dashboard* para este exemplo de aplicação.

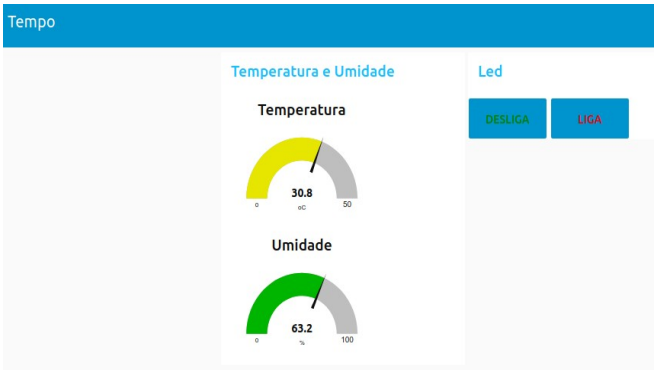

Fig. 12. *Dashboard* no Node-RED.

Caso o Node-RED esteja instalado em uma plataforma em nuvem, acessível via endereço IP público, a *dashboard* pode ser utilizada para controlar remotamente uma aplicação, por exemplo, utilizando um *smartphone*.

G. *Conclusão sobre os laboratórios*

Os laboratórios com Arduíno são interessantes para pequenos protótipos ou testes de conceito, em particular pela facilidade de uso proporcionada pelo Arduíno. Na interação com o Node-RED o uso do protocolo Firmata apresenta limitações. Funciona bem para para entradas e saídas digitais, sendo limitado para entradas analógicas, as quais podem gerar muitos valores que precisam ser manuseados pelo Node-RED. A alternativa mais interessante para interação direta do Node-RED com o Arduíno é o uso da comunicação serial.

Para dispositivos que possuem interface de rede, como o Arduíno com *shield* Ethernet ou o ESP com a interface *Wifi*, a comunicação com o Node-RED pode ser realizada usando um *broker* MQTT. Esta é uma opção moderna para as aplicações de Internet das Coisas.

O protocolo CoAP é interessante caso os dispositivos de Internet das Coisas precisem de interação direta de um usuário utilizando a Web e HTTP. Neste caso, o CoAP deve ser integrado ao HTTP por meio de um *proxy*.

Dentre os hardwares testados, cabe destaque para o ESP32, que possui possui interface *WiFi*, *Bluetooth* e vários sensores embutidos. Além de disponível em módulos dedicados a prototipagem, o ESP32 também é apropriado para implementação de soluções definitivas, em particular por seu baixo custo e pela facilidade de incorporá-lo aos demais dispositivos de hardware, como sensores e atuadores, em placas de circuito impresso.

No caso do Raspberry Pi, destaque para a possibilidade do mesmo ser utilizado como *gateway* ou na computação em névoa em aplicações de Internet das Coisas. Neste último caso, realizando o processamento de informações próximo aos dispositivos sensores e atuadores.

VI. CONCLUSÃO

Neste trabalho foram apresentados um conjunto de tecnologias, protocolos e ferramentas que podem ser utilizadas para a construção de protótipos e realizar provas de conceito para aplicações de Internet das Coisas. Foi realizado uma releitura das camadas de protocolos da Internet a luz das

novas aplicações de Internet das Coisas, analisando quais protocolos são mais adequados em cada situação, com destaque para o protocolo de aplicação MQTT.

Em termos de tecnologias e ferramentas para Internet das Coisas, foram apresentados e exemplificados diversos componentes de software, com destaque para a plataforma Node-RED, a qual permite interligar dispositivos físicos, funções para tratamento de dados e serviços em nuvem.

Outras ferramentas apresentadas foram o *broker* MQTT Mosquitto e os contêineres Docker. Estes últimos utilizados para implementar instâncias de software de forma independente do sistema operacional hospedeiro, o que facilita a criação de instâncias em sistemas de para computação em nuvem ou computação em névoa.

Também foram apresentados diversos dispositivos de hardware, os quais julgamos adequados para a construção de protótipos para Internet das Coisas, como as plataformas Arduíno, ESP8266 e ESP32 e o computador de placa única Raspberry Pi.

Da mesma forma que observamos nos últimos anos uma democratização no acesso às tecnologias de hardware, impulsionadas por plataformas didáticas como o Arduíno, estamos agora vivenciando uma democratização também ao desenvolvimento de software, com as plataformas *low code*. No caso específico das aplicações de Internet das Coisas, destacamos novamente a ferramenta Node-RED e seu poder para testar conceitos, incluindo tanto aspectos do tratamento de dados quanto da realização de interfaces usuário para os sistemas.

REFERÊNCIAS

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

[2] WOLLSCHLAEGER, Martin; SAUTER, Thilo and JASPERNEITE, Jürgen. The Future of Industrial Communication: Automation Networks in the Era of the Internet of Things and Industry 4.0, **IEEE Industrial Electronics Magazine**, march 2017.

[3] RATURI, Ankita and BUCKMASTER, Dennis. Connected Cows: Growing Plants , Raising Animals, and Feeding Communities through Connected Agriculture: An IoT Challenge, **IEEE IoT Magazine**, December, 2019.

[4] RAYES, Ammar and SALAM, Samer. **Internet of Things From Hype to Reality**: The Road to Digitization, Springer, 2019.

[5] KUROSE, James F. and ROSS, Keith W. **Computer Networking**: A Top-Down Approach Featuring the Internet, Pearson, 2001.

[6] COSTA, Francis da. **Rethinking Internet of Things**: A scalable approach to connecting everything. Apress Open, 2013.

[7] COLINA, Antonio Liñán; VIVES, Alvaro; ZENNARO, Marco; BAGULA, Antoine and PIETROSEMOLI, Ermanno. **Internet of Things in Five Days**, Internet Archive, 2016.

[8] LAMPKIN, Valerie; LEONG, Weng Tat; OLIVERA, Leonardo; RAWAT, Sweta; SUBRAHMANYAM,Nagesh and XIANG. Rong. **Building Smarter Planet Solutions with MQTT and IBM WebSphere MQ Telemetry**, ibm.com/redbooks, 2012.

[9] **MQTT Essentials**: The Ultimate Kickstart For MQTT Beginners. Disponível em: https://www.hivemq.com/mqtt-essentials/. Acesso em: 27/11/2020.

[10] CANTÚ, Evandro. MQTT, Wiki do IFPR, Foz do Iguaçu. Disponível em: http://wiki.foz.ifpr.edu.br/wiki/index.php/MQTT. Acesso em: 27/11/2020.

[11] **Node-RED:** Low-code programming for event-driven applications. Disponível em: https://nodered.org/. Acesso em: 27/11/2020.

[12] **Eclipse Mosquitto**: An open source MQTT broker. Disponível em: https://mosquitto.org/. Acesso em: 27/11/2020.

[13] **Docker**: A standardized unit of software. Disponível em: https://www.docker.com/. Acesso em: 27/11/2020.

[14] O'LEARY, Nick. **Arduino Client for MQTT**, knolleary. Disponível em: https://github.com/knolleary/pubsubclient. Acesso em: 27/11/2020.

[15] **Arduino**: Language Reference. Disponível em: https://www.arduino.cc/reference. Acesso em: 27/11/2020.

[16] **Arduino core for the ESP32**. espressif/arduino-esp32. Disponível em: https://github.com/espressif/arduino-esp32. Acesso em: 27/11/2020.

[17] **CoAP client, server library for Arduino**. hirotakaster/CoAP-simple-library. Disponível em: https://github.com/hirotakaster/CoAP-simple-library. Acesso em: 27/11/2020.