TÉCNICO LISBOA

Laboratório da Internet das Coisas

João Gonçalo Vieira Saramago

Dissertação para obtenção do Grau de Mestre em

Engenharia Informática e de Computadores

Orientador: Prof. Alberto Manuel Ramos da Cunha

Júri

Presidente: Prof. João António Madeiras Pereira
Orientador: Prof. Alberto Manuel Ramos da Cunha
Vogal: Dr. Rui António Policarpo Duarte

Novembro 2021

Este trabalho foi criado com a linguagem tipográfica LaTeX
no ambiente Overleaf (www.overleaf.com).

Abstract

With the growing use of the Internet of Things in the most diverse areas of work and leisure, it is important to offer IST students the possibility of receiving education in this area. The universities most interested in this field already have labs with microcontrollers, small computers, and wireless networks to interconnect devices. Besides networks, several have IoT platforms to manage communications, devices, and access to their data. Thus, to create an IoT lab at IST it is important to implement one of these platforms. After analyzing the open-source options available, ThingsBoard was the one that proved to be superior and was chosen to be implemented in the lab. The options of microcontrollers to be used to communicate with the platform were analyzed, and the ESP32 was the selected one. For students with little to no experience with microcontrollers, a library was created to simplify their communication with ThingsBoard. In addition, sets of sensors, actuators, and one microcontroller, called SensorBoxes, were created for installation and demonstration at the IST campus. Performance tests showed that the platform is adequate for the intended use, although the server provided by IST, where it was installed, had insufficient resources to use the platform with some intensity. Finally, it was possible to adapt existing lab work where two students would have to establish wired communications between microcontrollers. Using the platform these communications could be established over the internet, being an added value in situations like the one experienced during the COVID-19 pandemic.

Keywords

Internet of Things; Laboratory; ThingsBoard; ESP32;

Resumo

Com o aumento do uso da Internet das Coisas nas mais variadas áreas de trabalho e lazer, torna-se importante oferecer aos alunos do IST a possibilidade de obterem formação nessa vertente. As universidades mais interessadas neste ramo já disponibilizam laboratórios com microcontroladores, pequenos computadores e redes sem fios para interligar estes dispositivos. Para além de redes, várias dispõe de plataformas de IoT para gerir as comunicações, os dispositivos e o acesso aos dados destes. Assim, para criar um laboratório de IoT no IST é importante a implementação de uma destas plataformas. Após a análise das opções open-source disponíveis, o ThingsBoard foi a que se mostrou superior, tendo sido a escolhida para implementar no laboratório. Foram analisados os microcontroladores a utilizar para comunicar com a plataforma, tendo-se escolhido o ESP32. Para os alunos com pouca ou nenhuma experiência com microcontroladores, foi criada uma biblioteca que simplifica as comunicações destes com o ThingsBoard. Além disso, foram feitas montagens com sensores, atuadores e microcontroladores, chamadas SensorBoxes, para instalação e demonstração no campus do IST. Os testes de desempenho mostraram que a plataforma é adequada para o uso pretendido, embora o servidor disponibilizado pelo IST, onde esta foi instalada, tivesse poucos recursos para ser utilizado com alguma intensidade. Por fim, foi possível adaptar trabalhos de laboratório já existentes onde dois alunos teriam de estabelecer comunicações com fios entre microcontroladores. Com recurso à plataforma estas comunicações podem ser através da internet, sendo uma mais valia em situações como a vivida durante a pandemia do COVID-19.

Palavras Chave

Internet das Coisas; Laboratório; ThingsBoard; ESP32;

Conteúdo

Lista de Figuras

Lista de Tabelas

Acrónimos

ACIC	Aplicações e Computação para a Internet das Coisas
ADC	Analog to Digital Converter (Conversor analógico-digital)
AI	Ambientes Inteligentes
API	Application Programming Interface (Interface de programação de aplicação
CMU	Computação Móvel e Ubíqua
CoAP	Constrained Application Protocol (Protocolo de Aplicação Restrita)
DC	Direct Current (Corrente direta)
DIIC	Design de Interação para a Internet das Coisas
eCO2	Dióxido de carbono estimado
EEPROM	Electrically-Erasable Programmable Read-Only Memory
GPS	Global Positioning System (Sistema de posicionamento global)
HTTP	Hypertext Transfer Protocol (Protocolo de Transferência de Hipertexto)
IDE	Integrated Development Environment (Ambiente de Desenvolvimento Integrado)
ICSP	In-Circuit Serial Programming
IoT	Internet of Things (Internet das Coisas)
IP	Internet Protocol (Protocolo da Internet)
IST	Instituto Superior Técnico
I2C	Inter-Integrated Circuit
LED	Light-Emitting Diode (Diodo emissor de luz)

MEIC Mestrado em Engenharia Informática e de Computadores

MQTT Message Queuing Telemetry Transport

OS Operating System (Sistema Operativo)

OTA Over-The-Air (Remota)

PIR Passive Infrared (Infravermelho Passivo)

PVP Preço de Venda ao Público

REST Representational State Transfer (Transferência Representacional de Estado)

RGB Red Green Blue (Vermelho Verde Azul)

RNL Rede das Novas Licenciaturas

RPC Remote Procedure Call (Chamada remota de procedimento)

SDK Software Development Kit (Kit de desenvolvimento de software)

SPI Serial Peripheral Interface (Interface de Serial Periférica)

SSID Service Set Identifier (Identificador do conjunto de serviços)

TVOC Total Volatile Organic Compounds (Total de compostos orgânicos voláteis)

URL Uniform Resource Locator (Localizador uniforme de recursos)

1

Introdução

Conteúdo

Este capítulo apresenta o significado da Internet das Coisas, o porquê de querer realizar um trabalho nessa área e o que pretendo atingir com ele.

1.1 Internet das Coisas

A Internet das Coisas é o conceito em que diferentes tipos de objetos se conectam à Internet, conseguem comunicar entre si e ser controlados remotamente. Podem ser, entre muitas coisas, objetos banais como ténis que contam passos, sinais de trânsito inteligentes que favorecem a circulação nos trajetos mais movimentados ou contentores do lixo públicos que avisam a empresa responsável quando estão a ficar cheios e deve ser feita a recolha. É a ligação entre o mundo digital e o mundo físico. Tem como objetivo facilitar e otimizar a vida das pessoas automatizando diferentes tipos de cenários, tanto pessoais como profissionais. A Internet das Coisas é mais conhecida pelo seu nome inglês, *Internet of Things*, ou pela sua abreviação também inglesa, IoT.

1.2 Plataforma de IoT

A Internet das Coisas precisa de conectar dispositivos remotamente. A plataforma de IoT permite interligar e gerir dispositivos como sensores e atuadores, guardar, visualizar e processar dados recolhidos, podendo também controlar os acessos a estes dados e outras lógicas relacionas com a área de aplicação. Um dos muitos casos de uso pode ser gerir grandes frotas de veículos através da plataforma, para a qual os dispositivos com GPS estarão a enviar os seus dados de localização. Uma aplicação móvel de IoT direcionada ao utilizador, por exemplo para controlar uma casa inteligente, vai buscar à plataforma de IoT os dados dos sensores em tempo real ou do passado e controla os atuadores da casa. Podem também ser criadas automatizações na plataforma para que a casa acione atuadores de forma autónoma consoante os dados dos sensores. É uma ferramenta importante em diversas áreas de engenharia, tanto para otimização de processos utilizando dados de sensores para se perceber o que acontece ao longo do tempo, como para a criação de novas ideias e produtos para colocar no mercado e facilitar ou acrescentar algo à vida dos consumidores. Deste modo é algo que os alunos, especialmente na área de informática, devem ficar a conhecer durante a sua formação académica.

1.3 Contexto e Motivação

A motivação principal deste trabalho é começar a criar a infraestrutura para um Laboratório de IoT no Instituto Superior Técnico (IST). Trazer a possibilidade de gerir sensores e atuadores conectados à

internet através de uma plataforma de IoT. Permitir que os alunos possam aceder a sensores espalhados pelo campus e utilizá-los na realização de trabalhos e projetos de várias disciplinas. Este acesso poderá ser feito dentro ou fora do IST, podendo assim os alunos em qualquer parte do mundo aceder a esta rede de sensores e deste modo continuar os seus trabalhos. Para além do acesso aos dispositivos já existentes, os alunos também poderão criar os seus próprios e os integrar na rede. Uma infraestrutura destas traz novas possibilidades para várias disciplinas e cursos, nomeadamente no Mestrado de Engenharia Informática e de Computadores (MEIC):

- **Aplicações e Computação para a Internet das Coisas (ACIC)** [1] - No laboratório final da cadeira os alunos devem interligar dois controladores e estabelecer comunicação entre eles, normalmente isto era feito através do I2C, um tipo de comunicação por cabo. No entanto, devido à pandemia do COVID-19, no ano letivo de 2020/2021 já foi pedido que os alunos estabelecessem a comunicação entre controladores através da internet, tendo assim os alunos de estudar as opções possíveis e escolher uma, como por exemplo, usar um servidor MQTT. Através do sistema que se pretende colocar em prática o aluno conseguirá também criar a comunicação entre os dois dispositivos e será possível os gerir através da plataforma, tendo acesso ao histórico e a outros tipos de manipulação de dados, além de a comunicação ser realizada com maior segurança. Ao mesmo tempo o aluno estará a ter uma ideia mais realista sobre como funciona o mundo do IoT, ao ganhar conhecimento sobre como funcionam as plataformas de IoT usadas no mundo profissional para gerir sensores e atuadores.

- **Design de Interação para a Internet das Coisas (DIIC)** [2] - Os alunos podem integrar dados de sensores já existentes no IST nos seus projetos, trazendo-lhes assim novas possibilidades. Podem ainda criar os seus próprios sensores ou atuadores e os colocar em sítios relevantes para o trabalho que estão a desenvolver, desde que tenham conexão à internet.

- **Ambientes Inteligentes (AI)** [3] - Tal como em DIIC, mas com projetos num âmbito diferente, os alunos podem criar os seus projetos *low-level* montando controladores com sensores e atuadores, ou podem usar a rede de sensores existente para criar projetos *high-level* onde utilizam apenas software para aceder aos valores disponíveis na plataforma.

- **Computação Móvel e Ubíqua (CMU)** [4] - Abre a possibilidade de criar uma aplicação móvel baseada em dados de uma rede de sensores do IST, quer seja em tempo real ou dados guardados. Podem obter mais informações sobre o estado de várias zonas no IST e assim são criadas novas possibilidades de aplicações que não poderiam ser antes exploradas.

- **Dissertação de Mestrado** - Permite aos alunos que desejem realizar dissertações na área da Internet das Coisas usarem uma plataforma de IoT de forma gratuita alojada pelo IST. Pode ser

utilizada como base para a construção das suas ideias, como por exemplo, a criação de um sistema de controlo de qualidade do ar no interior do campus. As possibilidades de utilização são infindáveis e deste modo os alunos não precisam de se preocupar com a escolha e implementação de uma plataforma de IoT, podendo focar-se unicamente no desenvolvimento das suas ideias.

Apesar de estar bastante orientado para a Engenharia Informática devido aos conhecimentos necessários para a utilização deste sistema. Muitos cursos no IST já tem alguma componente de programação no seu currículo devido à influência da informática em todos os campos de trabalho atualmente. Deste modo, outros cursos podem usufruir também desta infraestrutura. Um curso de Engenharia Civil pode querer montar vários sensores em edifícios para realizar estudos, como por exemplo as aplicações estudadas em [5], tendo também já sido aplicado em salas de aula [6]. Um aluno de Engenharia do Ambiente pode realizar estudos sobre impactos ambientais recolhendo dados 24/7 e podendo mais tarde os analisar na plataforma ou os descarregar, como por exemplo a monitorização da qualidade da água [7]. Alunos de Engenharia e Gestão Industrial podem desenvolver projetos para monitorizar linhas de produção, trabalhar de acordo com a Industria 4.0 [8]. Ainda outro exemplo podem ser alunos de Engenharia Química para estudar ambientes laboratoriais, o comportamento ao longo do tempo de soluções ou trabalhos que estejam a realizar, por exemplo a monitorização da purificação da água [9]. Muitas possibilidades existem em todas as áreas e a disponibilização de uma plataforma destas no IST traria a oportunidade aos alunos de todos os cursos experimentarem o que não é só o futuro mas também o presente. Colocando assim os alunos do IST na vanguarda digital ao entrar no mundo profissional.

2

Estado da Arte

Conteúdo

Este capítulo apresenta o estado atual dos laboratórios de IoT no mundo académico, as principais tecnologias utilizadas neste trabalho e a razão que as levou a serem necessárias.

2.1 Laboratórios de IoT no ambiente académico

Com o aumento do uso da Internet das Coisas nas mais variadas áreas de trabalho e lazer, cada vez mais universidades procuram introduzir os seus alunos a esta área. Com esse objetivo, estas criam laboratórios de IoT virados à exploração tecnológica da área, onde os alunos exploram a tecnologia por detrás das comunicações para IoT, e laboratórios de IoT virados para a exploração aplicacional, onde os alunos procuram usar a tecnologia existente ou a melhorar para criar novos produtos relacionados com o IoT para uso do consumidor, trazendo assim maior conforto e segurança à sua vida. Com a criação destes laboratórios as universidades conseguem trazer um ambiente mais adequado para os seus alunos realizarem os projetos dos semestres e para desenvolvimento de teses. Além dos laboratórios, as faculdades permitem ainda que outras partes dos edifícios sejam automatizadas, como por exemplo luzes, deste modo os alunos além de explorarem a área também a conseguem meter em prática no mundo real, perto deles [10].

Um dos pontos principais que os laboratórios criados pelas outras universidades têm em comum é a disponibilização de um vasto leque de *hardware* para ser usado pelos alunos. Isto contempla diversos microcontroladores, como Arduinos e ESP32, e pequenos computadores de desenvolvimento, tanto baseados em ARM, sendo o Raspberry Pi o exemplo mais comum, como baseados em x86 [11]. Também existem variados tipos de sensores e atuadores para que os alunos consigam pôr em prática os seus projetos. Para auxiliar à compreensão e à resolução de problemas durante a realização de projetos, é comum existir no laboratório osciloscópios, multímetros e fontes de alimentação, havendo também alguns ferro de soldar circuitos para quando os projetos assim o requerem [12].

Para os alunos que pretendam criar projetos de IoT sem terem conhecimentos de eletrónica, apenas conhecimentos de programação, existe o lado mais aplicacional destes laboratórios. Uma das áreas a que esta parte dos laboratórios é mais direcionada são as casas inteligentes, isso reflete-se também no tipo de sensores e atuadores disponibilizados e nos tipos de projetos que lá são feitos. São disponibilizadas tecnologias já existentes e usadas no dia-a-dia pelo consumidor final, como Philips Hue e Samsung SmartThings [13] [14] [15], entre outros. Estes produtos já têm APIs desenvolvidas para que qualquer um com conhecimento em programação os consiga controlar. Existe outra alternativa ao uso de produtos comerciais, pode ser criada uma rede de sensores, que uma vez que esteja a funcionar estará sempre a enviar os dados dos seus sensores para um servidor, sendo este último depois capaz de partilhar os dados guardados ou em tempo real através da sua API. Para este tipo de funcionalidades usam-se as plataformas de IoT. Dos laboratórios que recorrem a esta tecnologia para interligar

dispositivos, alguns optam por versões pagas e de renome como as plataformas de IoT da Amazon, Microsoft ou Google, ou então optam por plataformas de patrocinadores dos seus laboratórios o que influência a escolha, no entanto há quem também se direcione para opções *open-source*, que para o trabalho a implementar é o mais adequado. Os laboratórios que não usam plataformas de IoT para gerir as comunicações normalmente disponibilizam apenas uma rede de comunicações, por exemplo a LoRaWAN [16], onde os dispositivos conseguem comunicar entre si, no entanto para as mensagens serem geridas e guardadas terá de ser o utilizador a criar essa funcionalidade na sua própria aplicação.

Outras áreas de foco destes laboratórios são o IoT para vestir [17], a segurança para peões, condutores e passageiros no mundo urbano [18] e a sustentabilidade e uso razoável de recursos, otimizando, por exemplo, o uso de ar condicionado, aquecimento e ventilação [15]. Isto tudo pode ser feito com recurso a plataformas de IoT.

2.2 Plataformas de IoT

No inicio da realização deste trabalho começou-se por explorar plataformas de desenvolvimento de *backend* de aplicações, tendo-se experimentado trabalhar com o Kuzzle [19], instalado o mesmo e usado as suas APIs. No entanto, rapidamente se percebeu que não era ideal para o trabalho e viu-se as vantagens de trabalhar diretamente com uma plataforma de IoT. Esta já tem as funcionalidades necessárias para gestão de dispositivos e níveis de privilégios adequados, entre muitas outras interessantes num ambiente de IoT. A própria empresa por detrás do projeto Kuzzle percebeu as vantagens e criou uma plataforma de IoT a partir da sua plataforma de *backend*. Esta plataforma de IoT foi disponibilizada após se ter desistido de utilizar o Kuzzle, depois de Abril de 2021.

Para escolher a plataforma de IoT a ser usada foram definidos vários pontos importantes:

- *Open-source*
- Tamanho da comunidade
- Desenvolvimento ativo
- Boa documentação
- SDKs para diversas plataformas
- Painel de controlo/administração
- Suporte a instalação local

Foram excluídas desde o inicio plataformas pagas ou com planos gratuitos limitados, como AWS IoT [20], Google Cloud IoT [21], Arduino IoT Cloud [22], entre outras conhecidas. Um *software* ser *open-source* traz várias vantagens como, não existirem custos para o utilizar, existir maior privacidade pois

qualquer pessoa pode ver o que o código está a fazer com os seus dados não havendo assim usos indesejados, haver uma maior flexibilidade no sentido de um desenvolvedor de código poder acrescentar funcionalidades que precise e ainda não estejam implementadas, não estar dependente da existência e interesses de uma empresa, entre outras. Tendo em conta que um software *open-source* é criado pela sua comunidade é bastante importante que a plataforma de IoT escolhida tenha uma grande comunidade por detrás dela e uma boa documentação, para que todos se consigam entender e utilizar todo o potencial das suas funcionalidades.

Entre as várias opções disponíveis para plataforma de IoT as que mais se sobressaíram, tendo em conta os critérios definidos, foram as seguintes:

- ThingsBoard [23]
- Kaa IoT [24]
- OpenRemote [25]
- SiteWhere [26]

2.2.1 Comparação das plataformas de IoT

Analisando melhor as hipóteses selecionadas, através da Figura 2.1 podemos verificar que o ThingsBoard, mesmo sendo a plataforma de IoT mais recente, lançada em dezembro de 2016, é de longe a mais popular, tendo o maior número de *Stars* no seu repositório do GitHub. O número de *Stars* de um repositório pode ser correlacionado com a sua popularidade. Em segundo lugar fica a plataforma Kaa com aproximadamente 14% do número de *Stars* do ThingsBoard, em terceiro o SiteWhere e por último o OpenRemote. Podemos ainda verificar através da mesma figura que a tendência de crescimento do ThingsBoard tem sido muito maior,

Observando a atividade dos repositórios nos 31 dias anteriores a 14 de setembro de 2021, verificamos que, tal como na comparação anterior, o ThingsBoard se destaca bastante dos outros com 158 *commits*, enquanto que o Kaa tem 8, o OpenRemote 55 e o SiteWhere 0. Todas as plataformas em questão têm painel de controlo para administrar dispositivos, utilizadores, visualizar dados, entre outras funcionalidades.

No passado, já outras pessoas compararam as plataformas de IoT *open-source* existentes. Leonidas Otalora comparou em maio de 2019 as várias opções da altura em termos de funcionalidades disponíveis e de integração das plataformas com outros dispositivos, tendo já nessa altura chegado à conclusão que o ThingsBoard era a melhor plataforma a utilizar, destacando-se em todos os pontos de comparação, entre eles a usabilidade e a documentação disponível [27]. Eduard Bitencourt e Willian

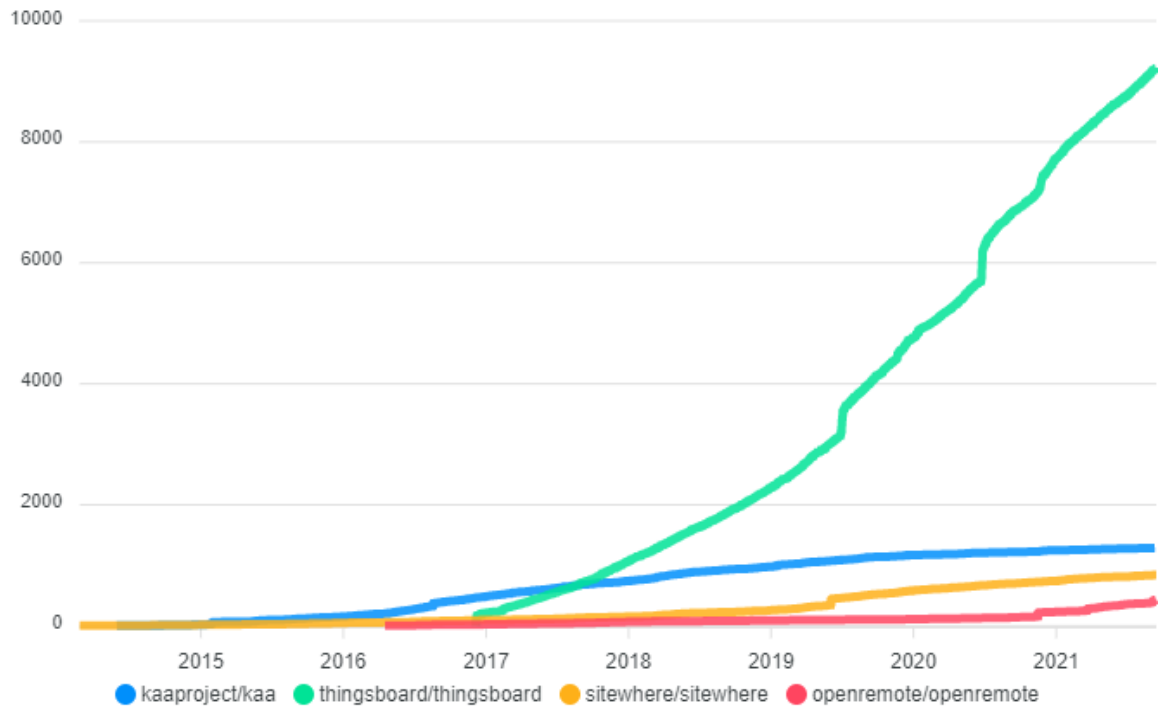


Figura 2.1: Número de *Stars* nos repositórios do GitHub das plataformas de IoT a 14/09/2021.

Anjos, em agosto de 2018, também desenvolveram um projeto com base no ThingsBoard, tendo comparado o mesmo com outras plataformas e chegado à conclusão que era de facto a melhor plataforma *open-source* a usar [28].

O facto de o *website* do ThingsBoard [23] indicar que várias empresas conhecidas são utilizadoras da sua plataforma mostra a sua maturidade e que é a melhor escolha, respeitando os critérios, para a aproximação deste trabalho a um ambiente profissional. Entre as várias empresas encontramos a T-Mobile, Bosch, Prosegur e a INSYS.

2.2.2 ThingsBoard

O ThingsBoard [23] é uma plataforma de IoT *open-source*. Permite interligar e gerir dispositivos como sensores e atuadores, guardar, visualizar e processar dados recolhidos, possibilitando criar novas aplicações com base nesses dados. É desenhado para ser escalável, tolerante a falhas, robusto, eficiente, personalizável e durável. Os dispositivos comunicam diretamente com a plataforma usando MQTT, HTTP ou CoAP, no entanto é possível usar *gateways* para traduzir outros protocolos para MQTT e serem interpretados pelo sistema, como Modbus, OPC-UA e LoRa. Neste trabalho o protocolo de comunicação usado pelos dispositivos será o MQTT, enquanto que a aplicação de computador usará HTTP para aceder à REST API. Os utilizadores da plataforma estão organizados em 3 níveis de

privilégios:

- O **Administrador do Sistema** é o responsável pelo servidor e empresas que utilizam o sistema para alojar os seus dispositivos.
- O **Locatário(*Tenant*)** é equiparado a uma unidade de negócio, empresa ou individual, que tem ou produz dispositivos. Pode ter múltiplas contas de administrador do seu negócio e milhões de clientes, dispositivos e ativos.
- O **Cliente** é o consumidor final, pode ser também uma empresa ou individual que comprou o acesso aos dispositivos. Capaz de aceder aos seus painéis de controlo e de visualizar ou aceder a dados de dispositivos autorizados. Pode ter múltiplas contas de acesso e milhões de dispositivos e ativos.

Existe a versão gratuita (Edição Comunidade) e a paga (Edição Paga) no entanto a versão gratuita tem todas as funcionalidades necessárias para este trabalho e um dos objetivos é usar ferramentas *open-source* gratuitas. A edição paga traz vantagens apenas para casos de usos mais avançados onde, por exemplo, seja preciso um sistema de privilégios mais evoluído, ou onde a pessoa responsável pelo sistema queira vender o sistema como seu e queira colocar o seu logo e tema nas páginas de administração. Existe ainda a versão Nuvem, é apenas a versão paga mas alojada pela própria empresa e com possibilidade de usar um domínio personalizado. Pode ser visto na Figura 2.2 a comparação entre as três versões, retirada da própria página na internet em inglês.

2.3 Microcontroladores em IoT

Com a necessidade de tornar muitos objetos inteligentes e os colocar nos mais variados cenários, como uma casa, um jardim público ou um terreno agrícola, deparamos-nos com diversos problemas. Numa casa o acesso à rede elétrica na maioria das vezes não é uma dificuldade, no entanto no meio de um jardim ou terreno agrícola isso nem sempre é uma possibilidade, assim é importante que estes dispositivos sejam o mais eficientes e de baixo consumo possível de modo a poderem ser alimentados viavelmente por baterias. Com a possibilidade de tornar quase tudo inteligente, muitos dispositivos serão necessários em diferentes localizações, assim para além da eficiência energética também é preciso que os próprios dispositivos sejam de baixo custo e em muitas situações que sejam invisíveis ao ser humano, ou seja, dispositivos pequenos que possam ser integrados nos próprios objetos. Assim pequenos microcontroladores são os dispositivos ideais a usar em IoT devido ao seu formato em que tudo está incluído num chip, com pequeno consumo energético e baixo custo de produção face aos microprocessadores. Certamente pequenos computadores têm melhor desempenho, no entanto para ler apenas os dados de alguns sensores e os enviar para a Internet pouco poder de processamento é

Figura 2.2: Comparação entre versões do ThingsBoard, retirada da sua página na internet (Inglês)

necessário e um microcontrolador capaz de se ligar à internet chega. Uma vez os dados na Internet, um único computador pode tratar os dados de muitos dispositivos.

Os pontos importantes na escolha de microcontroladores para IoT num ambiente académico são:

- **Baixo custo** - Uma vez que é para ser utilizado por pessoas com pouca experiência é mais provável que aconteçam erros e danos no dispositivo, assim para ser viável para o IST o custo de cada um tem de ser baixo.

- **Versatilidade** - De modo a diminuir a curva de aprendizagem o SDK do microcontrolador deve ser simples de usar no entanto também incluir funcionalidades mais avançadas para os alunos que o desejarem. Quanto mais funcionalidades o dispositivo possuir, melhor o investimento do IST e maior o leque de projetos possíveis de executar pelos alunos.

- **Popularidade** - Para facilitar a aprendizagem é importante haver o máximo de documentação disponível na internet para os alunos acederem de modo a esclarecer dúvidas ou a aprenderem a usar as funcionalidades pretendidas, para isso, além da documentação oficial, quanto maior a comunidade a usar o microcontrolador maior a disponibilidade de tutoriais e projetos em que este é usado.

Tendo em conta os pontos acima e após explorar na internet os tutoriais e projetos de IoT disponíveis, os seguintes microcontroladores foram os que se destacaram:

- Arduino Uno com placa Wi-Fi baseada num ESP8266
- ESP8266
- ESP32

2.3.1 Arduino Uno

O Arduino UNO R3 [29] é uma placa de desenvolvimento bastante popular baseada no microcontrolador ATmega328 e feita pela empresa Arduino [30]. Tem 14 pinos digitais de entrada/saída (dos quais 6 podem ser utilizadas como saídas PWM), 6 entradas analógicas, um cristal oscilador 16 MHz, uma ligação USB, uma ficha de alimentação de 7-12V, ICSP, e um botão reset. Tem uma tensão operacional de 5V e 32KB de memória flash. Não precisa de ser montada numa breadboard e existe uma grande variedade de placas que podem ser montados sobre este para lhe acrescentar funcionalidades.

Figura 2.3: Arduino Uno R3

2.3.1.A Placa Wi-Fi ESP8266 para Arduino Uno

Como mencionado anteriormente, existe uma grande variedade de placas para montar no Arduino Uno de modo a lhe acrescentar funcionalidades. Uma funcionalidade muito desejada no mundo do IoT é a possibilidade de ligação ao Wi-Fi. Para esse fim existem placas com o ESP8266 [31] onde o Arduino consegue, através de Serial ou SPI, pedir comunicações com internet sem ter de se sobrecarregar com a gestão da própria conexão ao Wi-Fi.

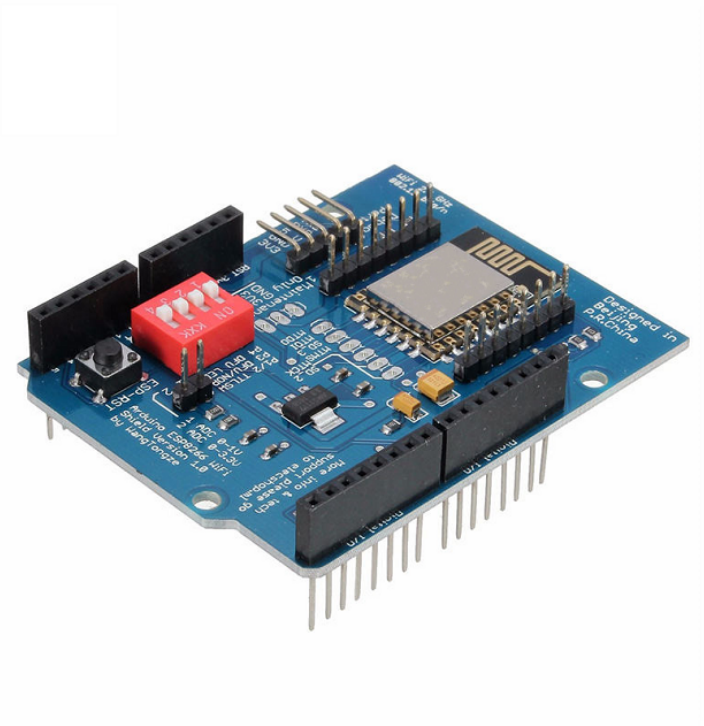


Figura 2.4: Placa Wi-Fi ESP8266 para Arduino Uno

2.3.2 ESP8266

O ESP8266 [32] é um microcontrolador da empresa Espressif [33] baseada num CPU de 32 bits RISC com um núcleo a 80MHz ou 160MHz e comunicação Wi-Fi. Lançado pela primeira vez em 2014 já passou por várias versões e existe em diversos formatos, desde muito compacto como o Wemos D1 Mini [34], até versões idênticas ao formato do Arduino Uno. Tem um SDK baseado em FreeRTOS e outro sem OS, cada um com as suas vantagens e desvantagens. Veio mais tarde a ser substituído pelo ESP32, sendo no entanto ainda produzido atualmente. O que o tornou tão popular na altura foi o facto de o seu baixo preço ser idêntico ao de microcontroladores sem Wi-Fi e por ser compatível com o Arduino IDE [35]. As especificações mudam consoante a versão do ESP8266 em causa e a placa de desenvolvimento em que for montado.

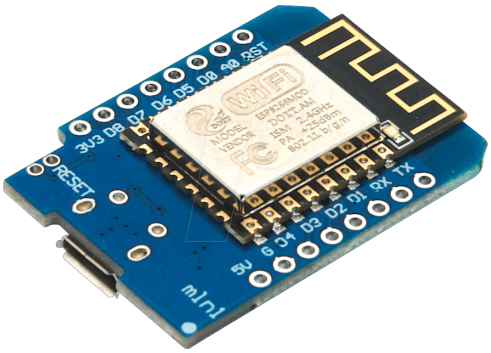


Figura 2.5: ESP8266 Wemos D1 Mini

2.3.3 ESP32

O ESP32 [36], sucessor do ESP8266, é um microcontrolador da empresa Espressif [33] baseada num CPU de 32 bits com dois núcleos a 160MHz ou 240MHz, comunicações Wi-Fi e Bluetooth. Lançado pela primeira vez em 2016, já passou por várias versões e, tal como o seu antecessor, também existe em vários formatos. É uma atualização bastante significativa ao ESP8266 tanto em desempenho como em pinos disponíveis e o que eles podem fazer. O seu SDK é baseado em FreeRTOS o que permite usar as suas funcionalidades no desenvolvimento de programas. É um dos mais populares microcontroladores de baixo custo para IoT e é compatível com o Arduino IDE [35]. As especificações, como do Wi-Fi e Bluetooth, mudam consoante a versão do ESP32 e a placa de desenvolvimento em que for montado.

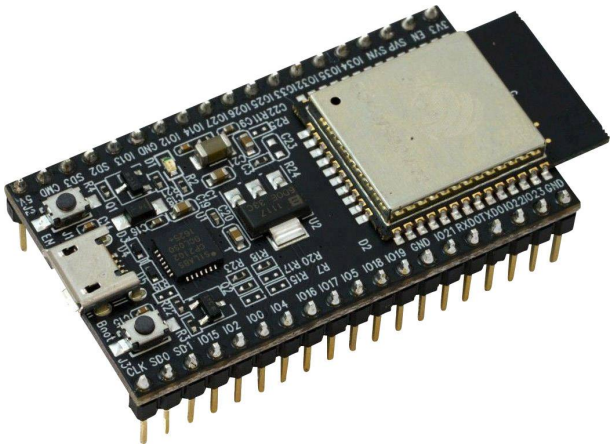


Figura 2.6: ESP32 DevKitC V4

2.3.4 Comparação dos microcontroladores

Em termos de custos, o Arduino Uno oficial tem como PVP 20€, sendo possível o encontrar em desconto em algumas lojas, as versões clones podem se encontrar por volta de 6€ vindas da China com portes para Portugal incluídos. Placas Wi-Fi baseadas no ESP8266 encontram-se à venda por volta de

4,5€ vindas da China com portes incluídos para Portugal. O ESP8266 encontra-se atualmente à venda por aproximadamente 3€ vindo da China com portes incluídos para Portugal. Já o ESP32 encontra-se à venda por aproximadamente 5€ vindo da China e com portes incluídos para Portugal.

O Arduino Uno é um dispositivo bastante popular entre iniciantes a microcontroladores pela sua simplicidade e tolerância a abusos, no entanto não possui acesso ao Wi-Fi, assim pessoas que já o possuem e querem ganhar essa funcionalidade por vezes optam por lhe acrescentar uma placa Wi-Fi, deste modo continuam a usar um dispositivo que já conhecem. O ESP8266 e ESP32 são também bastante populares, normalmente as pessoas começam por entrar no mundo dos microcontroladores através dos Arduinos e depois sim mudam para estes dispositivos pelas suas funcionalidades extra. No entanto isto não quer dizer que sejam dispositivos mais difíceis de utilizar, fora a configuração inicial necessária no Arduino IDE o resto é praticamente idêntico, sendo que as principais funcionalidades existentes no Arduino Uno também existem nestes dois e se usam da mesma maneira. Atualmente a escolha de um ESP8266 ao invés de um ESP32, deve-se maioritariamente à redução de custos pois o ESP32 é superior em todos os outros aspetos, daí ser o sucessor.

A vantagem do Arduino Uno com a placa Wi-Fi, neste trabalho, deve-se ao facto de o IST já possuir estes dispositivos. No entanto, quando for ocasião de adquirir mais microcontroladores, o ESP32 apresenta-se como a melhor opção pois, comparando com o Arduino Uno, é muito mais rápido e já integra o Wi-Fi no próprio controlador, sendo também mais barato, por volta de metade do preço quando comparado com o conjunto do Arduino e a placa Wi-Fi. Face ao ESP8266, o ESP32 é mais caro no entanto é mais rápido e mais recente, sendo o sucessor o foco da empresa passou a estar nele. Além disso o ESP32 já traz Bluetooth, trazendo assim um maior leque de possibilidades aos alunos. Por isto, o ESP32 foi o microcontrolador escolhido para este trabalho, usando-se os outros para comparação e para compatibilidade caso algum aluno ainda assim decida optar por usar algum deles.

2.4 FreeRTOS

Com a necessidade do microcontrolador ler sensores e enviar os seus dados em diferentes intervalos de tempo, dos atuadores terem de ser manipulados e ainda de ser mantida uma conexão à espera de novas mensagens, é necessário executar todas estas tarefas de modo preemptivo. Assim conseguimos respeitar os períodos de envio de dados dos sensores enquanto podemos receber mensagens sem ter de esperar por um ciclo de envio de dados. Tornamos assim o sistema mais responsivo para por exemplo acionar os atuadores. Se um atuador for um LED que pisca a uma determinada frequência, conseguimos interromper a tarefa que estiver a ser executada no momento em que é necessário mudar o estado do LED, respeitando assim o mais possível a frequência desejada, sem com isso bloquear a execução de outras tarefas no microcontrolador ou complicar bastante mais o código. É também

possível definir prioridades nas tarefas, assim as prioritárias nunca serão interrompidas por outras de prioridade inferior. Para tornar isto possível usamos o FreeRTOS [37], um sistema operativo em tempo real para microcontroladores. Além de tornar o anterior possível, ainda facilita a organização do código comparando com a criação de um código que implementasse uma lógica similar manualmente, sendo maior a vantagem quanto maior for o código a executar. No ESP32 o FreeRTOS está instalado de forma nativa pois o seu SDK já se baseia nele e por ter dois núcleos de processamento consegue realizar duas tarefas em simultâneo. No ESP8266 temos a opção de usar um SDK com o sistema operativo FreeRTOS ou uma versão sem sistema operativo. Para ser possível ter o FreeRTOS no Arduino Uno temos de o instalar como biblioteca [38]. Compilando o código vazio sem e com a biblioteca importada acresce 7126 bytes, passando a memória de 1% ocupada para 23%. Podemos ver assim que mesmo em microcontroladores com pouca memória é possível usar o FreeRTOS.

2.5 Docker

O Docker [39] é uma plataforma que torna mais fácil, simples e seguro criar, lançar e gerir contentores. Os contentores estão isolados entre eles e têm o seu próprio software e configuração. Basicamente um contentor pode conter um sistema operativo, digamos Ubuntu [40], com programas instalados como o ThingsBoard e deste modo é possível correr uma instância do ThingsBoard em qualquer computador ou servidor usando esse contentor.

3

Arquitetura

Conteúdo

Este capítulo apresenta a arquitetura do trabalho implementado, capacidades da plataforma de IoT e a integração desta com os sensores e atuadores.

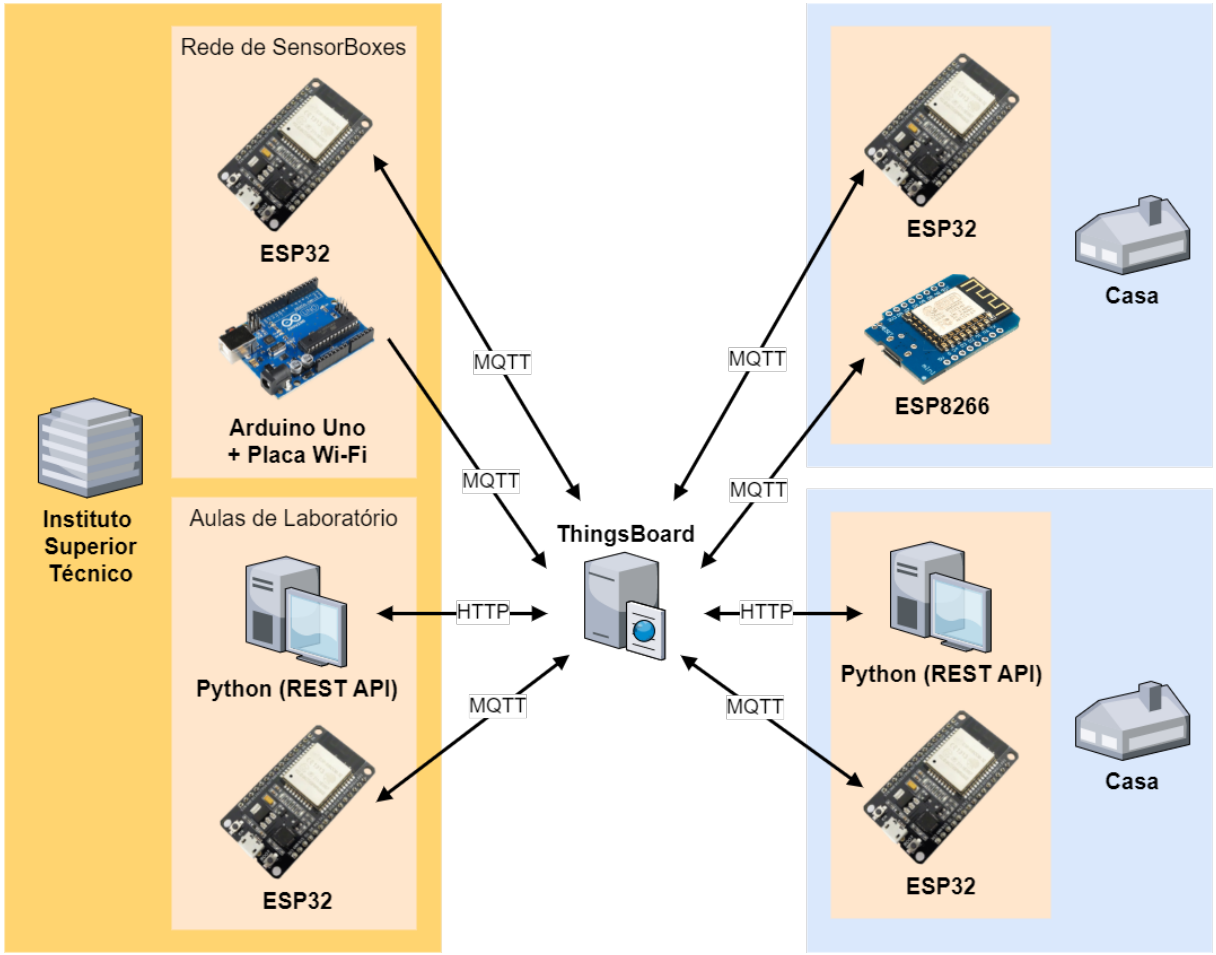


Figura 3.1: Arquitetura do trabalho a implementar.

Este trabalho pretende criar um ambiente de Internet das Coisas no IST e dar uma base aos alunos para explorarem esta área nas diferentes disciplinas e cursos existentes. Para isso ser possível foi implementada uma plataforma de IoT, neste caso o ThingsBoard, num servidor acessível a todos através da internet.

3.1 Plataforma de IoT

A plataforma serve de ponto de ligação entre todos os microcontroladores e o utilizador, seja este último através da página de administração ou da REST API, por exemplo, o utilizador pode através da REST API receber os dados de sensores de determinados microcontroladores e de acordo com a sua lógica acionar os atuadores de outros.

Na página de administração o utilizador pode gerir os seus dispositivos e analisar os dados por eles transmitidos em tempo real ou consultando o histórico. Esta análise pode ser feita recorrendo a painéis de visualização onde podem ser criados gráficos e tabelas com os dados, e também botões para controlar atuadores. O utilizador pode definir, através de cadeias de regras e por tipos de dispositivos, o que acontece às mensagens que chegam ao servidor, se são descartadas, se devem ser tratadas antes de as guardar, se ativam alarmes, entre muitas outras opções. Os dispositivos podem ainda ser associados a ativos, por exemplo, relacionar dispositivos com os edifícios ou objetos onde estão instalados.

Existem três níveis de privilégios na plataforma, o administrador de sistema que gere os locatários e as definições do sistema, os administradores de locatário que gerem os dispositivos e ativos do seu locatário definindo também quem são os seus clientes e a que dispositivos eles têm acesso, e por fim o nível dos clientes que têm acesso aos dispositivos que lhes forem permitidos e aos seus painéis de visualização. Um aluno que queira apenas aceder aos sensores existentes no campus terá que pedir uma conta de cliente a um administrador do locatário(*tenant*) que gere esses dispositivos. Também é possível tornar os dispositivos públicos, no entanto para evitar usos mal intencionados ou que vão contra o uso esperado, é preferível que quando algum aluno ou grupo os queira usar tenha de pedir dados de acesso. Caso o aluno queira criar os seus próprios dispositivos, poderá o fazer pedindo uma conta de locatário, assim age como dono do seu negócio e fornecedor de acesso a dispositivos aos seus clientes. Caso o aluno queira instalar o seu dispositivo no campus, disponível para todos os alunos, terá que pedir a um administrador da conta que gere os dispositivos do IST para adicionar o seu lá, ou então terá de ser ativada uma das técnica de provisionamento para que o aluno sozinho consiga adicionar o seu dispositivo à rede no IST.

Aos dados dos sensores enviados para o ThingsBoard chama-se telemetria. Os atributos são pares chave-valor sobre informação do dispositivo, pode ser o que o utilizador entender, como versão de *firmware*, parâmetros ou configurações. O dispositivo consegue tanto publicar como ler os seus atributos. Os utilizadores também podem aceder às funcionalidades das páginas de administração através da REST API recorrendo a pedidos HTTP, estes podem ser abstraídos utilizando SDKs em várias linguagens de programação já disponibilizados pelo ThingsBoard.

3.2 Biblioteca para microcontroladores

Tendo em conta que o sistema é para ser usado maioritariamente por alunos que acabam de ser introduzidos aos microcontroladores é importante que a utilização deste seja o mais simples possível, assim é criada uma biblioteca para os microcontroladores que simplifique a conexão e comunicação destes com a plataforma. O ThingsBoard tem uma biblioteca oficial para Arduino que comunica com a

sua plataforma de IoT através da API de MQTT. Para simplificar ainda mais a conexão e comunicação com o servidor para os alunos sem experiência, é criada uma biblioteca, extensão da anterior, que agiliza o uso das seguintes funcionalidades:

- Conexão à rede Wi-Fi, incluindo rede Eduroam [41]
- Conexão ao servidor
- Manutenção das conexões
- Envio de detalhes da SensorBox
- Subscrição a comandos para os atuadores
- Provisionamento de dispositivos
- Atualizações OTA

A biblioteca é compatível com o Arduino Uno e a sua placa Wi-Fi, o ESP8266 e o ESP32. O protocolo usado para comunicar com a plataforma é o MQTT, no entanto o utilizador não terá a necessidade de conhecer a estrutura das mensagens usadas pelas diversas funcionalidades tendo em conta que pode chamar funções que já tratam desse tipo de coisas. Com esta biblioteca o mesmo ficheiro de código pode ser usado nos três microcontroladores mencionados.

3.3 Montagens de microcontroladores e sensores

Para demonstrar o uso da plataforma aos alunos e para que os que não estejam interessados em montar circuitos com sensores e atuadores possam ainda assim usufruir da mesma, serão criados dispositivos para espalhar no IST, deste modo será possível obter medições do estado do campus através destes. A estas montagens dá-se o nome de SensorBoxes.

A SensorBox é o nome dado ao conjunto de sensores, atuadores e microcontrolador pronto a colocar em qualquer lado desde que tenha acesso à internet e à eletricidade. Nesta montagem também existem LEDs para mostrar o estado do código e ainda uma maneira de ligar uma fonte de alimentação, por exemplo um transformador de 5V. As SensorBoxes serão para uso interior e existem em duas versões, uma maior para demonstração em que existe um LED para cada sensor e atuador de modo a ser possível ver quando estes enviam ou recebem mensagens, e outra versão menor apenas usada para ser colocada numa parede ou mesa a recolher informações. Os microcontroladores ESP32, ESP8266 e Arduino Uno com a placa Wi-Fi são compatíveis com a construção da SensorBox. A escolha de atuadores deve ser pensada de acordo com o público alvo do dispositivo, se é para ser usado por uma pessoa ou pelos alunos em geral. Os atuadores influenciam o ambiente onde eles estão, por

exemplo, uma campainha quando acionada vai emitir barulho e este pode ser indesejado no local onde se encontra.

4

Implementação

Conteúdo

Este capítulo explica a implementação do trabalho desenvolvido, as dificuldades e limitações encontradas.

Os três principais componentes do sistema são:

- Plataforma de IoT (ThingsBoard [23])
- Biblioteca para Microcontroladores
- Sensores e Atuadores (SensorBox)

Neste trabalho a plataforma de IoT foi instalada num computador pessoal para desenvolvimento e depois em três servidores diferentes para experimentar um ambiente de produção. É explicado como utilizar as páginas de administração e a REST API. Relativamente à biblioteca criada, são mencionadas as suas dependências externas de *software* e *hardware*, as dificuldades encontradas ao utilizar o Arduino Uno com a placa Wi-Fi fornecida pelo IST, o ambiente de desenvolvimento e como a utilizar a biblioteca. Sobre as SensorBoxes é explicada a sua construção, a sua compatibilidade com os microcontroladores e sensores, as duas versões criadas e o seu diferente código.

4.1 ThingsBoard

4.1.1 Configuração

Graças à boa documentação do ThingsBoard é fácil correr o seu servidor e pode ser instalado de diversas maneiras em várias plataformas. A versão utilizada neste trabalho foi a 3.3.1.

4.1.1.A Ambiente de desenvolvimento

Para o desenvolvimento da maior parte deste trabalho foi usado o Docker a correr o servidor ThingsBoard num computador pessoal, tanto em Windows como em MacOS. Com este tipo de instalação todo o ambiente em que o servidor é executado está isolado e assim é fácil de o correr ou eliminar sem deixar dependências instaladas no computador. Uma vez o Docker instalado, o servidor colocado a funcionar no Windows seguindo este tutorial [42] e no MacOS seguindo este [43]. Nesta etapa do trabalho decidiu-se utilizar a versão com a base de dados PostgresSQL e para serviço de fila de mensagens escolheu-se a ”Em Memória”. O uso do Docker com as configurações escolhidas para o servidor é a maneira mais simples de ter o ThingsBoard a correr e pronto a desenvolver o restante trabalho.

4.1.1.B Ambiente de produção

Para aproximar este trabalho de um uso real experimentou-se instalar o ThingsBoard diretamente em servidores. Todos eles a correr Ubuntu Server 20.04 LTS, assim o processo de instalação foi maiorita-

riamente igual para todos, seguindo este tutorial [44]. Começou-se por instalar num computador Intel NUC com 8GB de RAM e processador i5 de 8ª geração, o i5-8259U, instalado como servidor numa rede local mas sem ser possível aceder a este através do exterior. Usando configurações idênticas ao ambiente de desenvolvimento, excepto no uso de Kafka como serviço de fila de mensagens por ser o recomendado para ambientes de produção, não foram apresentadas dificuldades na instalação nem no uso do painel de administração web ou conexão de dispositivos ao servidor.

Com o objetivo de fazer uma instalação do ThingsBoard acessível através da internet foi experimentado a fazer num servidor t2.micro da AWS EC2, que usa 1 vCPU e tem 1GB de RAM. Segui-se o tutorial acima mencionado, tendo-se antes aberto os portos necessários [45]. Estes recursos apresentaram-se insuficientes para usar o serviço de fila de mensagens Kafka, devido à reduzida quantidade de RAM. Voltou-se a usar a configuração idêntica ao ambiente de desenvolvimento com parcial sucesso. Devido ao método de gestão de recursos da instância t2.micro, a disponibilidade do CPU era gerida através de créditos, ou seja, era possível usar mais poder de processamento do que o anunciado pela instância, no entanto os créditos seriam gastos mais rápido do que eram repostos. Uma vez todos os créditos gastos o poder de processamento ficava bastante limitado, muito inferior ao desempenho de 1 vCPU, deste modo a execução do ThingsBoard ficava comprometida, demasiado lento para processar pedidos tanto de acesso ao painel de administração como de comunicações com microcontroladores. O resultado final eram alguns minutos de funcionamento correto do servidor seguidos por incapacidade de o usar até que a instância fosse reiniciada. O lado positivo a retirar desta experiência foi ter sido possível instalar o ThingsBoard de modo a ficar acessível por todos na internet.

Para conseguir replicar a instalação feita no AWS t2.micro mas num servidor com mais recursos e sem gastar dinheiro foi então pedido à RNL, uma rede de apoio aos cursos de informática do IST, uma máquina virtual com Ubuntu Server 20.04 LTS, 2 vCPU e 2GB de RAM e os mesmos portos abertos que na da AWS. Uma vez a máquina disponibilizada, foi feita a instalação do ThingsBoard com PostgresSQL e Kafka. Inicialmente o servidor funcionava corretamente mas rapidamente o uso de RAM ficou saturado e apesar de a comunicação com os dispositivos continuar a funcionar corretamente, o painel de administração na internet ficou bastante lento. Voltou-se então a usar como serviço de fila de mensagens a opção ”Em Memória”e deste modo já foi possível correr o servidor sem problemas. Apesar de não ser o serviço recomendado para produção, é suficiente para a realização deste trabalho e para usos leves. O conjunto de configurações usadas já deverá ser capaz de gerir algumas centenas de mensagens por segundo através de MQTT. O servidor ficou acessível através do endereço thingsboard.rnl.tecnico.ulisboa.pt, sem a necessidade de estar a aceder ao ThingsBoard através de um IP.

4.1.2 Níveis de Privilégios

Existe o administrador do sistema que é quem consegue criar novos locatários e administradores de cada locatário. Abaixo deste temos os administradores de locatário, podemos ter neste caso um locatário chamado Instituto Superior Técnico e como administradores as pessoas responsáveis por gerir os dispositivos instalados, por exemplo, a rede de SensorBoxes do IST. Estes poderão, entre muitas outras coisas, criar os dispositivos e gerir quem tem acesso aos mesmos. Os alunos que queiram ter acesso às SensorBoxes do IST terão que pedir uma conta de cliente do locatário do IST. Deste modo é possível controlar quem tem acesso aos dados dos dispositivos. Caso um aluno queira criar os seus próprios dispositivos e os integrar com a plataforma terá de pedir uma conta como administrador do seu próprio locatário, assim agirá como empresa que tem os seus dispositivos e que pode fornecer o acesso aos dados destes aos seus clientes.

4.1.3 Utilizar Página de Administração

As instruções sobre como utilizar a página de administração podem ser encontradas na Wiki do repositório [46]. Cada tipo de utilizador verá uma página de administração adequada ao seu nível de privilégios, devido às diferentes funcionalidades a que cada um tem acesso.

4.1.3.A Criar locatários e os seus administradores

O administrador do sistema pode criar novos locatários iniciando sessão na sua conta e indo para a página dos locatários (*Tenants*). No parte superior direita da página existe um símbolo de mais, clicando nele aparece um diálogo para preencher os dados do novo locatário, como se pode ver na fig. 4.1.

Neste diálogo apenas é obrigatório preencher o título, o perfil de locatário já está selecionado por defeito e fora algum uso especifico não precisa de ser alterado.

Após adicionar o locatário é preciso criar contas para os seus administradores. Para isso deve-se clicar em cima do locatário e selecionar a opção ”Gerir administradores de locatários”. Depois disso o processo é idêntico ao de adicionar um locatário, clica-se no símbolo mais e introduz-se o e-mail desejado. No final do diálogo deve-se escolher o método de ativação, ou seja, como é que administrador deve criar a sua palavra-passe. É possível visualizar logo o endereço para aceder à página de criação da palavra-passe ou o enviar para o endereço de e-mail inserido, como pode ser visto na fig. 4.2.

Depois de adicionado pelo menos um administrador já é possível gerir o locatário acedendo à sua página de administração.

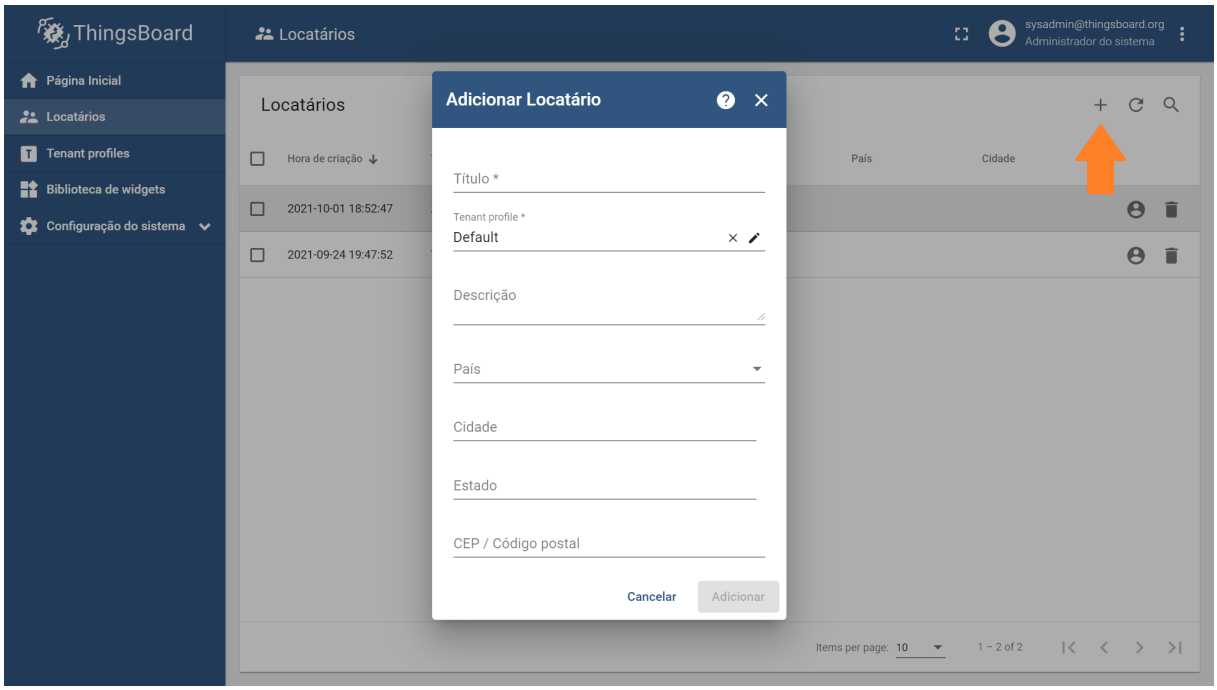


Figura 4.1: Diálogo para adicionar novo locatário.

4.1.3.B Criar clientes e os seus usuários

O processo de criação de clientes e contas para os seus usuários é muito parecido com o de criação de locatários e os seus administradores. Esta criação faz-se iniciando sessão na conta de um administrador do locatário desejado. Acede-se à página de clientes e clica-se no símbolo mais, insere-se pelo menos o titulo no formulário. Uma vez o cliente adicionado clica-se nele e na página que aparece clica-se em ”Gerir usuários”. Ao clicar para adicionar um novo usuário apenas o e-mail é obrigatório e no fim deve-se escolher como é apresentado o endereço da página para criação da palavra-passe.

4.1.3.C Criar dispositivos

Um administrador de locatário pode criar novos dispositivos para a sua empresa. Para isso deve aceder, na sua conta, à página ”Dispositivos”e clicar no símbolo mais. No diálogo que aparece apenas precisa de escolher o nome do dispositivo. Recomenda-se que sejam adicionadas credenciais para evitar que outras pessoas se consigam fazer passar por este dispositivo, por exemplo, neste trabalho é utilizado o *token* de acesso. Assim para o dispositivo se conectar ao servidor tem de fornecer as suas credenciais de modo a provar a sua identidade.

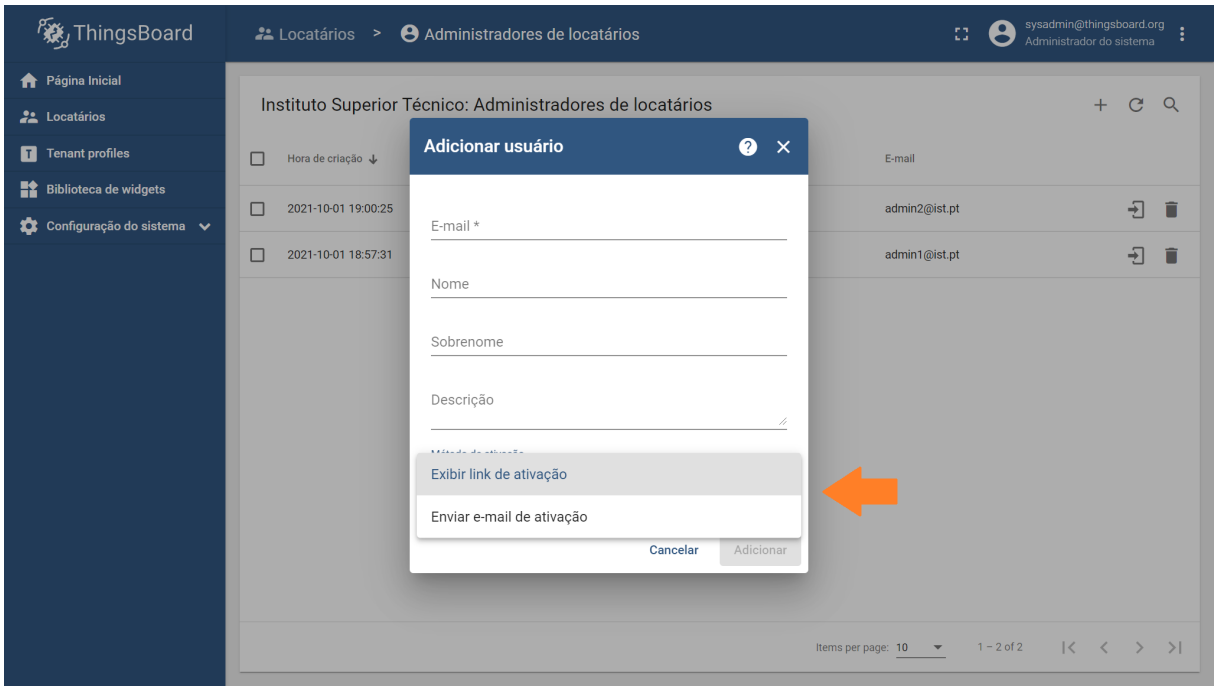


Figura 4.2: Diálogo para adicionar novo administrador do locatário.

4.1.3.D Provisionamento

Para ser possível adicionar dispositivos à plataforma sem ter de os inserir manualmente na página de administração podemos ativar uma técnica de provisionamento de dispositivos. Existem duas possíveis, sendo uma delas permitir a criação de dispositivos pré-provisionados, ou seja, uma empresa fabrica dispositivos e obtém uma lista de IDs únicos de cada um, depois pode adicioná-los em conjunto no ThingsBoard sem ficarem provisionados. Uma vez que um destes dispositivos se tente ligar ao servidor, este confirma se já existe algum registado com o seu ID único mas ainda por provisionar, caso seja verdade este regista-se obtendo credenciais de acesso. A outra técnica é idêntica à primeira mas sem estar limitado a uma lista de dispositivos, consiste em simplesmente deixar todos os dispositivos que tenham as chaves de provisionamento se registarem no ThingsBoard. Esta última opção é a mais adequada para o caso de uso deste trabalho, onde os alunos precisam de adicionar os seus dispositivos, pessoais ou do IST, ao servidor. Com esta última técnica, para se adicionar dispositivos à rede de SensorBoxes do IST já não se teria de pedir a uma administrador para adicionar o dispositivo manualmente, este apenas fornecia as chaves de provisionamento à pessoa em questão e ela conseguiria adicionar o seu próprio dispositivo. No entanto, tal como se fosse adicionado manualmente, a pessoa ao não ser administradora do locatário IST não terá acesso à pagina de administração do dispositivo, apenas à visualização de dados como cliente.

Para se ativar o provisionamento de dispositivos deve-se ir à secção de perfis de dispositivos. Cri-

ando um novo perfil ou selecionando um já existente, ao editar encontramos uma secção chamada provisionamento de dispositivos. Nesta podemos escolher entre as duas técnicas ou desativado. Ao escolher qualquer uma das técnicas aparecerão duas chaves de provisionamento que devem ser partilhadas com quem queira adicionar os seus dispositivos ao servidor, como é possível ver na fig. 4.3.

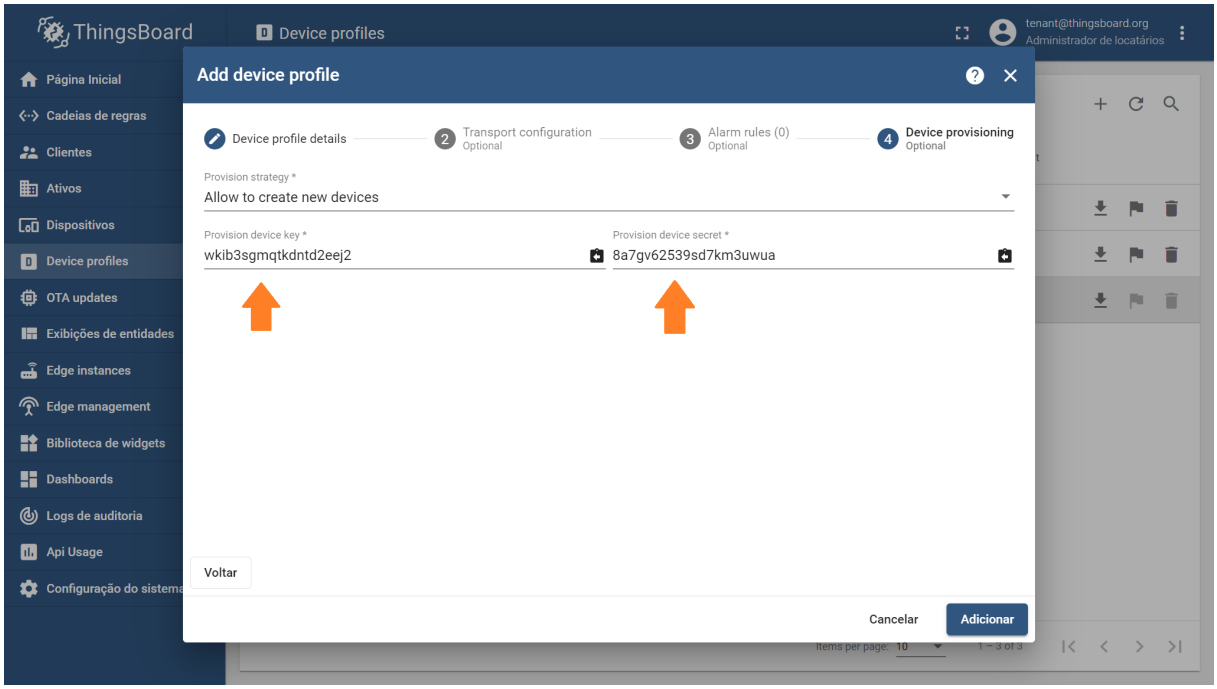


Figura 4.3: Ativar provisionamento de dispositivos.

Para pré-provisionar dispositivos deve-se ir à página de gestão de dispositivos e quando se clica em adicionar, em vez de se selecionar adicionar novo dispositivo, deve-se escolher importar dispositivo, o que permite inserir ficheiros com listas de dispositivos.

4.1.3.E Atualizar remotamente

Uma funcionalidade bastante útil para o caso das SensorBoxes é a possibilidade de atualizar o seu *firmware* remotamente. Estando elas montadas em diversos sítios do IST, ir atualizar uma a uma, caso fosse preciso, seria trabalhoso. Deste modo apenas é necessário carregar o ficheiro com a atualização para o ThingsBoard e relacionar os dispositivos com os seus ficheiros. Caso existam várias SensorBoxes iguais poderá ser criado um perfil de dispositivo que as identifique e como a mesma atualização deve ser feita em todas elas basta atribuir a atualização ao perfil e automaticamente todos os dispositivos associados a esse perfil serão atualizados.

Para carregar o ficheiro de uma atualização para o servidor deve-se ir à página ”Atualizações OTA”e clicar no símbolo mais. O título e a versão devem ser os mesmo que foram definidos no ficheiro de código da atualização. Escolhe-se o perfil de dispositivos para qual a atualização fica disponível, não

querendo dizer que todos sejam atualizados, apenas que só esse tipo de dispositivos poderá optar por fazer a atualização. Finalmente o ficheiro binários pode ser carregado no servidor ou pode ser fornecido um URL para a localização do ficheiro na internet.

Uma vez a atualização de *firmware* criada pode se ir à página de perfis de dispositivos atribui-la a todos os dispositivos do perfil, ou ir à pagina dos dispositivos e atribuir individualmente a cada um. Para isso basta clicar em cima do dispositivo desejado, clicar no botão, com o símbolo de um lápis, para editar e depois selecionar a atualização desejada no campo referente ao *firmware* atribuído, como pode ser visto na fig. 4.4.

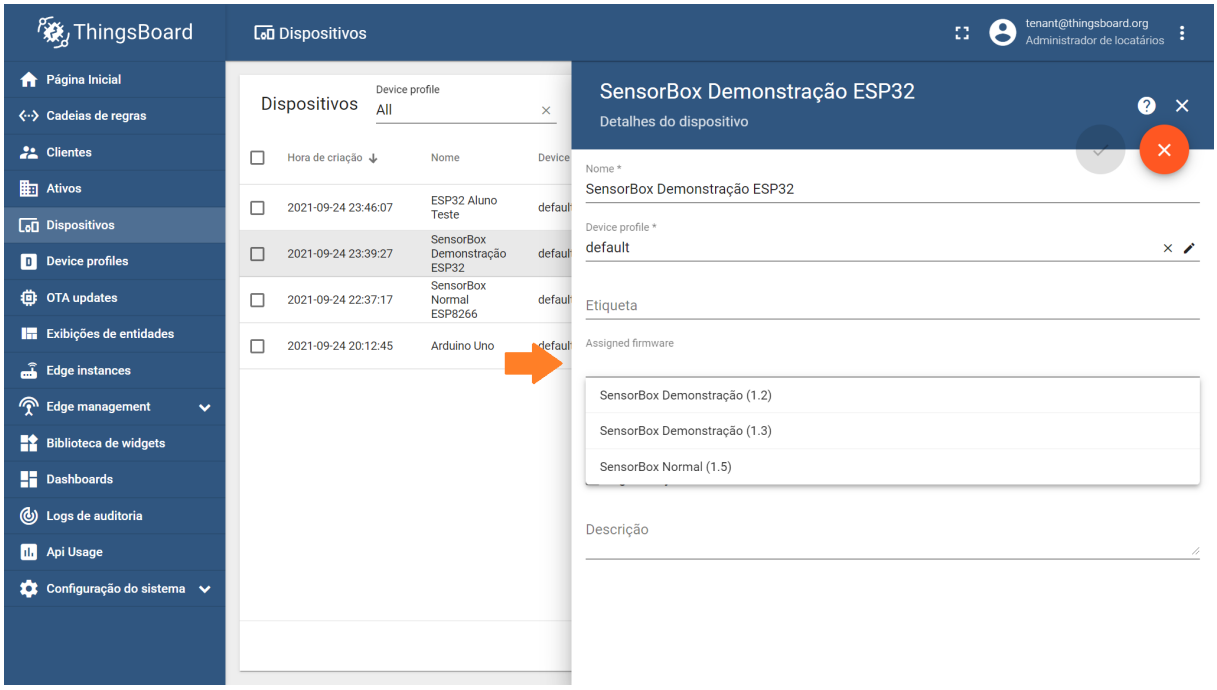


Figura 4.4: Atribuir atualização remota a um dispositivo.

4.1.4 Utilizar API

As instruções sobre como utilizar a REST API podem ser encontradas na Wiki do repositório [46]. Para se interagir, através de uma aplicação, com o servidor ThingsBoard e os seus dispositivos usa-se a REST API [47]. Esta pode ser usada diretamente através de pedidos HTTP ou pode ser usada com recurso a bibliotecas criadas em várias linguagens de programação que abstraem os pedidos HTTP [48]. A opção escolhida neste trabalho foi utilizar a biblioteca em Python [49] por simplificar o código necessário comparando com o uso direto da REST API e por Python ser das linguagens mais usadas pelos alunos do IST. Há que ter em atenção ao se usar a biblioteca pois esta está separada em funcionalidades da versão gratuita e as funcionalidades da versão paga, neste trabalho apenas é usada

a versão gratuita denominada por ”ce”, que significa *Community Edition* ou ”Edição da Comunidade”.

4.1.4.A Iniciar sessão

Depois de a biblioteca Python ”tb-rest-client”ter sido instalada no ambiente de desenvolvimento podemos iniciar sessão, como utilizador, da seguinte forma:

```
from tb_rest_client.rest_client_ce import *
from tb_rest_client.rest import ApiException

base_url = "thingsboard.rnl.tecnico.ulisboa.pt"
server_url = "https://" + base_url

username = "tenant@thingsboard.org"
password = "tenant"

with RestClientCE(base_url=server_url) as rest_client:
    try:
        rest_client.login(username=username, password=password)
        print("Inicio de sessao feito com sucesso!")

        ... # Executar novos pedidos a API aqui

    except ApiException as e:
        print(e)
```

Uma vez a sessão iniciada já dispomos do *token* necessário para nos identificarmos perante a API e assim já a podemos utilizar.

4.1.4.B Acionar atuadores

Após se iniciar sessão podemos acionar os atuadores com o seguinte código:

```
actuators_deviceID_str = "970a00b0-1d7f-11ec-a892-475609c76ca2"
actuators_deviceID = DeviceId('DEVICE', actuators_deviceID_str)

request = {
    "method": "setBuzzer",
    "params": {
        "value": True
    }
}
response = rest_client.handle_two_way_device_rpc_request(
    ↪ actuators_deviceID, request)
```

O ID do dispositivo pode ser obtido através da página do dispositivo. Este deve ser definido como o objeto DeviceID para poder ser usado na biblioteca, este representa a *string* com o ID e o tipo de entidade em questão, neste caso um dispositivo, mas poderia noutra situação referir-se a um ID de um

ativo. Depois de acordo com a documentação dos pedidos RPC formulamos o corpo do pedido a ser enviado, o nome do método RPC e os parâmetros a enviar.

4.1.4.C Receber dados de sensores

Para recebermos os dados dos sensores temos de abrir um WebSocket com o servidor e subscrever aos dispositivos que desejamos. Isto é algo que a biblioteca Python do ThingsBoard não faz e que teremos de fazer manualmente. Para isso instala-se a biblioteca "websocket". O código para subscrever a dados de sensores de dispositivos é o seguinte:

```
import json
import websocket
import _thread
import ssl
from tb_rest_client.rest_client_ce import *
from tb_rest_client.rest import ApiException

base_url = "thingsboard.rnl.tecnico.ulisboa.pt"
server_url = "https://" + base_url
username = "tenant@thingsboard.org"
password = "tenant"

sensors_deviceID_str = "970a00b0-1d7f-11ec-a892-475609c76ca2"

with RestClientCE(base_url=server_url) as rest_client:
    def on_message(ws, message):
        msg = json.loads(message)
        sensors_data = msg["data"]
        if not sensors_data:
            print(message)
            return
        if 'light_intensity' in sensors_data:
            intensity = sensors_data["light_intensity"][0][1]
            print("Intensidade da Luz:", intensity)

    def on_error(ws, error):
        print(error)

    def on_close(ws, close_status_code, close_msg):
        print("### WebSocket fechado ###")

    def on_open(ws):
        def run(*args):
            sub_cmd = {
                "tsSubCmds": [{
                    "entityType": "DEVICE",
                    "entityId": sensors_deviceID_str,
                    "scope": "LATEST_TELEMETRY",
                    "cmdId": 1
                }],
                "historyCmds": [],
                "attrSubCmds": []
            }
```

```
44              sub_cmd_str = json.dumps(sub_cmd)
45              ws.send(sub_cmd_str)
46              print("Subscrito aos sensores!")
47          _thread.start_new_thread(run, ())
48
49      try:
50          rest_client.login(username=username, password=password)
51          print("Inicio de sessao feito com sucesso!")
52          token = rest_client.configuration.api_key["X-Authorization"]
53          ws_url = "wss://" + base_url + "/api/ws/plugins/telemetry?token="
                ↪  + token
54          ws = websocket.WebSocketApp(ws_url, on_open=on_open, on_message=
                ↪ on_message, on_error=on_error, on_close=on_close)
55          ws.run_forever(sslopt={"cert_reqs": ssl.CERT_NONE})
56      except ApiException as e:
57          print(e)
```

Após o inicio de sessão guardamos o *token* recebido (linha 52) para conseguirmos abrir o WebSocket. Neste código abrimos um WebSocket seguro ”wss://”pois o servidor tem certificado TLS, caso contrário seria ”ws://”. No entanto, devido a um erro durante a verificação de certificado desativamos a verificação, caso contrário a solução seria descarregar o certificado e inseri-lo no código. Uma vez aberto é executada a função `on_open(...)`, onde criamos o comando de subscrição à telemetria do dispositivo desejado e enviamos. Assim sempre que for recebida uma mensagem será executada a função `on_message(...)` onde filtramos e verificamos os dados recebidos. Deve ser tido em consideração o tempo que demora a executar esta função, pois isso impede a receção de novas mensagens atempadamente. Assim é recomendado abrir novas *threads* para a manipulação dos dados recebidos.

4.2 Biblioteca SensorBox-ThingsBoard

4.2.1 Dependências externas

A biblioteca depende de *software* externo para o seu funcionamento, para que esta funcione é preciso cumprir os seguintes requisitos.

4.2.1.A Microcontroladores

A biblioteca foi criada com os microcontroladores ESP32, ESP8266 e Arduino Uno com placa Wi-Fi em mente. Assim, estes são os únicos que se pode dizer ser compatíveis e terem sido testados. O uso da biblioteca com qualquer outro microcontrolador não é garantido que funcione como descrito neste trabalho. O Arduino Uno tem as funcionalidades limitadas ao envio de telemetria e atributos.

Para utilizar o ESP32 e o ESP8266 no Arduino IDE é preciso lá instalar os seus SDKs, as versões utilizadas foram:

- ESP32 - Versão 1.0.6
- ESP8266 - Versão 2.7.4, a biblioteca oficial do ThingsBoard não é atualmente compatível com versões iguais ou superiores à 3.0.0.

4.2.1.B Bibliotecas

Tanto a biblioteca criada como a biblioteca oficial do ThingsBoard, em que esta se baseia, usam bibliotecas externas. Para garantir que a biblioteca funciona corretamente e como descrito neste trabalho é preciso usar as seguintes bibliotecas externas:

- Arduino ThingsBoard SDK - Esta é a biblioteca oficial do ThingsBoard. A versão mais atualizada no momento da escrita é a 0.5.0, no entanto esta tem um problema com a receção de comandos RPC, estão a tentar desserializar a informação quando não é preciso. Assim a versão usada neste trabalho é a deste commit [50] e deve ser descarregada do repositório e instalada como .zip no Arduino IDE. Uma vez lançada a nova versão da biblioteca já será possível a instalar a partir do gestor de bibliotecas do Arduino IDE com esta correção.
- PubSubClient - Versão 2.8.0
- ArduinoJson - Versão 6.18.4

No caso de ser usado o Arduino Uno é preciso instalar ainda as seguintes bibliotecas:

- ArduinoHttpClient - Versão 0.4.0
- WiFiEspAT - Versão 1.3.1

4.2.1.C Placa Wi-Fi para Arduino Uno

Durante a criação da biblioteca experimentou-se usar o Arduino Uno para receber comandos RPC, ou seja, receber comandos para atuadores. Foi usada a biblioteca WiFiEsp, uma das mais populares para integrar placas WiFi baseadas no ESP8266 com o Arduino. Devido ao tamanho das mensagens MQTT que o ThingsBoard envia para o Arduino, este não conseguia lidar com tantos dados a ser recebidos de uma só vez, a linha RX do Serial ficava cheia e não conseguia receber tudo. Com a biblioteca WiFiEspAT já é explorado o modo de receção passivo, introduzido a partir do firmware AT 1.7, o SDK NonOS 3 do ESP8266. Assim o Arduino já consegue controlar o fluxo de dados que recebe, já não recebe tudo de uma vez. Com esta biblioteca e uma redução de tamanho da biblioteca oficial para ganhar memória livre, o Arduino Uno já conseguiu receber comandos para os seus atuadores. No entanto modificar a biblioteca oficial é desvantajoso em termos de futuras actualizações e neste caso a maneira de reduzir o seu tamanho foi retirar os logs da biblioteca ficando também assim sem saber informações

caso algum problema inesperado aconteça com a execução do código da biblioteca. Na versão 0.5.0 da biblioteca oficial descartaram mesmo a opção de usar comandos RPC e outras funcionalidades que requeressem subscrição de mensagens pelo Arduino Uno pois não são adequadas a um microcontrolador com tão pouca memória. Assim a biblioteca criada neste trabalho tem as mesmas limitações que a oficial pois baseia-se nela. Manteve-se o uso da biblioteca WiFiEspAT pois, apesar de no final do trabalho já não ser necessária, continua a ser vantajosa pela sua flexibilidade caso o utilizador queira fazer as alterações necessárias à biblioteca oficial, assim ainda poderá usar a biblioteca da SensorBox para criar o seu código. Devido a esta experiência, o SDK da placa WiFi foi atualizado para o NonOS 3.0.4, a última versão do SDK sem sistema operativo pois a empresa focou-se a partir dai no SDK com FreeRTOS.

O SDK da placa Wi-Fi pode ser atualizado seguindo as instruções desta página [31] e usando o SDK descarregado deste repositório [51]. Há que ter em conta o diferente mapa das partições comparando com o do tutorial, ou seja, os diferentes endereços em que cada ficheiro deve ser escrito. A informação sobre os endereços a usar pode ser encontrada no manual de utilização do SDK em causa.

4.2.2 Ambiente de desenvolvimento

Para programar os microcontroladores foi escolhido o Arduino IDE [35] por ser um IDE simples de usar e dedicado à programação deste tipo de dispositivos. Este IDE, de origem, só traz o *software* necessário para programar microcontroladores da empresa Arduino. A versão do IDE utilizado foi a 1.8.10. Este trabalho foi desenvolvido em Windows e em MacOS.

4.2.2.A Instalar microcontroladores no Arduino IDE

Para se poder programar placas com o ESP32 ou com o ESP8266 no Arduino IDE é preciso instalar o *software* que saiba lidar com essas placas através do Gestor de Placas. Para instalar o ESP32 deve ser seguido este tutorial [52]. Instalar o ESP8266 é um processo quase idêntico, apenas muda o URL adicionado nas Preferências, como se pode ver neste tutorial [53]. Ao instalar a versão 2.7.4 do ESP8266 no MacOS 11.4 ocorre um erro ao tentar compilar o código para o microcontrolador. O erro encontrado está descrito como problema no repositório do *software* do ESP8266 para o Arduino IDE [54], a solução para o mesmo encontra-se na mesma publicação, onde referem que se deve substituir duas linhas na biblioteca PySerial.

4.2.2.B Instalar bibliotecas no Arduino IDE

As bibliotecas podem ser instaladas de duas formas, através do gestor de bibliotecas ou através de um ficheiro .zip. Para instalar a biblioteca da SensorBox terá de usar o ficheiro .zip pois a biblioteca não

se encontra listada no gestor de bibliotecas. Assim o ficheiro .zip deve ser descarregado do repositório [46] e instalado seguindo o tutorial adequado [55]. O tutorial indicado também mostra como instalar bibliotecas com o gestor de bibliotecas, útil para a instalação das dependências externas.

4.2.3 Ligação ao Wi-Fi

Esta biblioteca foi criada num ambiente de desenvolvimento com uma rede Wi-Fi que possui o modo de autenticação WPA2-PSK, a tradicional rede Wi-Fi residencial com uma SSID e uma palavra-passe pré-partilhada. No IST a rede Wi-Fi usada é a Eduroam, cujo o modo de autenticação é o WPA2 Enterprise, este tipo requer que o utilizador tenha um nome de utilizador registado e uma palavra-passe, portanto requer que a biblioteca use outro método de autenticação para se ligar à rede Wi-Fi. Apesar de não ter sido possível testar, foi adicionado ao ESP32 a funcionalidade de se conseguir conectar a redes com WPA2 Enterprise, para a implementar foram seguidos os exemplos deste repositório [56].

4.2.4 Falha de eletricidade e os atuadores

Ao se ligar uma SensorBox que tenha atuadores, estes vão ficar num estado que está predefinido no código. Ao longo do tempo é expectável que o estado dos atuadores seja alterado através de comandos RPC. Caso a SensorBox seja desligada e novamente ligada ou haja uma outra interrupção temporária da eletricidade, os atuadores voltam ao seu estado inicial, o predefinido no código, mesmo que este não seja o estado que estavam antes de faltar a eletricidade.

Seria possível guardar o estado de cada atuador como atributo no servidor, isto era, sempre que o microcontrolador recebesse um comando RPC para acionar um atuador seria também enviado o novo estado do atuador como atributo para o servidor. No entanto isto iria aumentar o tráfego de dados na rede e que o microcontrolador teria de enviar por cada comando RPC recebido. Outra opção seria guardar os estados dos atuadores na memória não volátil do microcontrolador, a EEPROM. O problema de a usar para guardar tais dados com alguma frequência é que esta memória tem um limite de escritas finito, no ESP32 normalmente assume-se 100.000 escritas, mas pode variar. Dependendo do caso de uso, a memória pode durar vários anos ou poucos dias.

Assim decidiu-se não implementar esta funcionalidade poupando tamanho da biblioteca e não estando a dar uma funcionalidade que pode estar a prejudicar o microcontrolador sem o utilizador ter essa noção. Um utilizador mais avançado que a deseje pode posteriormente a adicionar ao seu código seguindo a lógica que preferir.

4.2.5 Como utilizar

As instruções sobre como utilizar a biblioteca criada podem ser encontradas na Wiki do seu repositório [46].

Para se utilizar a biblioteca tem de se a incluir no inicio do ficheiro de código:

```
1 #include <SensorBoxTB.h>
```

4.2.5.A Iniciar e manter comunicações

O microcontrolador pode se ligar a redes com autenticação WPA2-PSK e WPA2 Enterprise, sendo que a última opção não foi testada por falta de acesso a uma rede deste tipo, como explicado anteriormente. Para iniciar a configuração das comunicações deve se definir globalmente o nome da rede Wi-Fi, a password, o servidor do ThingsBoard e o *token* de acesso do dispositivo:

```
1 char* ssid = "your-ssid";
2 char* password = "your-password";
3 char* thingsboard_server = "thingsboard_ip_or_hostname";
4 char* token = "device_token";
```

Para se ligar a uma rede WPA2 Enterprise, como a Eduroam, deve-se ainda definir o nome de utilizador:

```
1 char* username = "your-username";
```

Depois deve ser criado o objeto SensorBox, o termo usado pela biblioteca para definir o dispositivo com sensores e atuadores:

```
1 SensorBox sb(ssid, password, thingsboard_server, token);  // WPA2-PSK
2 // ou
3 SensorBox sb(ssid, username, password, thingsboard_server, token);  //
      WPA2 Enterprise
```

Para inicializar as comunicações, isto é, ligar à rede WiFi e ao servidor do ThingsBoard, usar dentro da função `setup()` o seguinte:

```
1 void setup() {
2   ...
3   sb.initComms();
4   ...
5 }
```

De modo a manter as comunicações abertas deve ser sempre executado na função `loop()` o seguinte:

```
1 void loop() {
2   ...
3   sb.loop();
4 }
```

4.2.5.B Publicar detalhes do dispositivo

Podemos publicar os detalhes dos dispositivos como atributos, por exemplo, informações como unidades das medições feitas e versão de *firmware* usado. Devemos definir globalmente uma lista de objetos DeviceDetails, este objeto é criado com duas listas de caracteres a representar a informação que se quer enviar. No seguinte exemplo, primeiro temos o nome do sensor e seguidamente a unidade dos valores que este envia.

```
1 DeviceDetails details[] = {
2   { "humidity", "%" },
3   { "motion",   "boolean" },
4   { "buzzer",   "boolean" },
5   ...
6 };
```

De forma a enviar estes dados para o servidor deve-se chamar a seguinte função após se inicializar as comunicações:

```
1 sb.sendDeviceDetails(details, COUNT_OF(details));
```

4.2.5.C Publicar dados de sensores

O envio de dados dos sensores é feito da seguinte forma para variáveis `float` ou `int`:

```
1 sb.sendTelemetryFloat("temperature", temperatureValue);
2 sb.sendTelemetryInt("humidity", humidityValue);
```

Mais tipos de variáveis podem ser enviadas usando os métodos adequados, podem ser consultados na documentação da biblioteca oficial do ThingsBoard.

4.2.5.D Receber comandos para atuadores

Para receber informações que manipulem atuadores, por exemplo, ligar um LED, é preciso estar subscrito a essas mensagens. Para isso o ThingsBoard usa RPC, chamada de procedimento remoto, isto é,

através da API do ThingsBoard é possível executar funções do microcontrolador e receber os resultados remotamente. As funções a serem executadas devem ser criadas da seguinte forma:

```
RPC_Response processSetLetIntensity(const RPC_Data &data) {
  // exemplo
  led_intensity = data["value"];
  ...
  return RPC_Response("led_intensity", led_intensity);
}
```

A variável `data` contém uma lista com a informação recebida do ThingsBoard, necessária para executar a função, neste caso o valor para a intensidade chama-se `value` mas poderia ter sido enviado com qualquer outro nome. No final da função devolvemos o resultado desta como se fosse uma variável, neste caso enviamos a variável `led_intensity` com o valor recebido para confirmar que a função foi corretamente executada e que foi lido o valor correto.

Para o dispositivo saber a que RPCs deve subscrever cria-se a lista com estes, sendo o primeiro argumento o nome público do processo que é chamado pela API e o segundo argumento é a função que é desencadeada por essa chamada:

```
RPC_Callback callbacks[] = {
  {"setLetIntensity", processSetLetIntensity},
};
```

Com fim a subscrever aos RPCs executa-se o seguinte método após se inicializar as comunicações:

```
sb.subscribeActuatorsCmds(callbacks, COUNT_OF(callbacks));
```

Uma vez isto feito, o microcontrolador estará subscrito às mensagens RPC e chamará as funções adequadas quando as receber.

4.2.5.E Provisionamento de dispositivos

Para fazer o provisionamento do dispositivo, assumindo que as configurações necessárias do servidor já estão feitas, deve-se inicializar o objeto `SensorBox` de outra maneira. Deve ser fornecido o nome desejado para o dispositivo, a chave de provisionamento e a palavra secreta de provisionamento. Ambas as últimas já devem ter sido configuradas no servidor.

```
char* device_name = "your-device_name";
char* provision_device_key = "provision-device-key";
char* provision_device_secret = "provision-device-secret";

SensorBox sb(ssid, password, thingsboard_server, device_name,
    provision_device_key, provision_device_secret);  //WPA2-PSK
```

```
// ou
SensorBox sb(ssid, username, password, thingsboard_server, device_name,
    provision_device_key, provision_device_secret);  //WPA2 Enterprise
```

Uma vez o provisionamento feito, o token do dispositivo será guardado na EEPROM, deste modo das próximas vezes que o microcontrolador for ligado usará logo o *token* em vez de tentar fazer novo provisionamento de um dispositivo que agora já existe. Caso se queira apagar o *token* da EEPROM deverá se chamar a seguinte função:

```
clearSavedToken();
```

Esta pode ser chamada num ficheiro de código de Arduino vazio, apenas importando a biblioteca SensorBox. Deste modo já se pode fazer um novo provisionamento sem o código tentar utilizar imediatamente o *token* que estava guardado.

4.2.5.F Atualizações Remotas (OTA)

Para ser possível atualizar remotamente o *firmware* do microcontrolador, ou seja, o código do programa que está a ser executado, é preciso que este subscreva às mensagens que o avisam sobre atualizações existentes. No código do microcontrolador deve estar definido o titulo do *firmware* e a sua versão, por exemplo:

```
#define CURRENT_FIRMWARE_TITLE    "Indoor Air Sensors FW"
#define CURRENT_FIRMWARE_VERSION  "1.0.2"
```

Depois das comunicações estarem inicializadas podemos subscrever às mensagens sobre atualizações:

```
sb.checkFirmwareUpdates(CURRENT_FIRMWARE_TITLE,CURRENT_FIRMWARE_VERSION);
```

Uma vez isto feito, sempre que o servidor disponibilizar novas atualizações para este dispositivo ele atualizará automaticamente. Há que ter em atenção ao atualizar dispositivos que corram no seu código alguma técnica de provisionamento e assim tenham na EEPROM o *token* guardado, este poderá ser apagado durante a atualização, caso isso aconteça, como o dispositivo já existe não se conseguirá provisionar novamente nem obter o seu *token*.

4.2.6 Código

O código pode ser visualizado no seu repositório do GitHub [46]. A explicação do código criado está comentada nos ficheiros da biblioteca. Exemplos de uso podem ser encontrados no mesmo repositório dentro da pasta ”examples”.

4.3 SensorBox

A SensorBox é o nome dado neste trabalho à montagens de sensores, atuadores e um microcontrolador que comunique com o ThingsBoard. Existem duas versões, a de demonstração e a normal. Para além das versões referidas é mencionado também o uso do kit do Arduino sozinho, este já inclui uma pequena base de montagem para o microcontrolador e uma *breadboard*, esta montagem não tem qualquer proteção por isso será apenas útil em espaços controlados ou fechados, como as vitrinas existentes no IST para afixação de folhas. O uso do kit do Arduino como SensorBox tem como única vantagem o facto do IST já possuir vários e ser assim mais fácil dedicar alguns para este uso.

4.3.1 Construção

A SensorBox foi pensada, nesta fase, para espaços interiores. Tem como objetivo ter todo o seu interior visível de modo a os alunos perceberem melhor com o que estão a trabalhar. Deve incluir LEDs que indiquem o seu estado de funcionamento e uma ficha para ligar uma fonte de alimentação. A construção deve ser o mais universal possível de modo a ser compatível com o máximo de sensores e atuadores. Para ser possível instalar sensores que façam medições sobre o ar, este deve conseguir circular, assim optou-se por criar uma construção aberta com o estilo sanduíche. Devido ao formato típico dos ESP32, as *breadboards* normais de 400 pinos são demasiado estreitas, assim normalmente encaixam se duas juntas montando-se o ESP32 ao meio. Devido a isto, a SensorBox deve ser capaz de acomodar duas *breadboards* juntas. O IST possui kits de Arduino Uno oficial [57], estes vêm com uma placa de montagem do Arduino com uma *breadboard*, esta placa tem dimensões próximas à montagem do ESP32 referida anteriormente, assim a SensorBox consegue ser compatível com as duas opções. Além dos microcontroladores e das *breadboards*, deve também ser possível montar sensores que já tenham o seu próprio circuito numa placa.

Tendo em conta o contexto académico em que se insere o trabalho decidiu-se criar duas versões da SensorBox, uma para demonstração e outra normal para ser apenas colocada no local em que se pretende fazer medições sem tanta preocupação em se perceber em detalhe o que está a acontecer. A versão de demonstração será apenas uma versão mais comprida da normal, de modo a ser possível instalar uma fita de LEDs endereçáveis com etiquetas, correspondendo cada LED ao envio ou receção de mensagens de cada sensor ou atuador, mais dois LEDs extra para mostrar o estado da ligação ao Wi-Fi e ao ThingsBoard. A versão normal, por não ter espaço para etiquetas terá apenas três LEDs montados na *breadboard*, um para o estado da ligação Wi-Fi, um para o estado da ligação ao ThingsBoard e outro para quando é recebido ou enviado qualquer mensagem relacionada com os sensores e atuadores.

Para a base de montagem foi escolhida a madeira pela sua fácil manipulação, sendo possível fa-

cilmente aparafusar sensores e atuadores ou outros materiais necessários à organização de cabos, evitando se assim, mas não inviabilizando, a colagem, que mais tarde poderia vir a falhar. Para proteger a montagem feita na base é usado uma placa de acrílico montada sobre quatro espaçadores. De modo a não inviabilizar o uso de sensores de ultrassom, foi cortado uma secção da placa de acrílico, onde devem então ficar montado este tipo de sensor. É possível ver o desenho das construções feitas no apêndice A.

Para alimentar o circuito decidiu-se usar um transformador de 5V com conector DC. Não se optou por alimentar pela porta USB pois aumentava a probabilidade de alguém mal intencionado roubar o transformador, visto que transformadores USB são o método mais comum de carregar a maioria dos dispositivos electrónicos hoje em dia. Outra razão é a compatibilidade, o Arduino Uno usa uma porta USB de formato diferente do ESP32, assim usando o transformador de 5V este pode ser logo ligado aos respetivos pinos de 5V de cada microcontrolador.

O desenho aberto da SensorBox trás a desvantagem de se acumular pó no seu interior ao longo do tempo, no entanto é preferível obter medições sobre o ar mais fiáveis e quando for necessário soprar o pó. Outra desvantagem de ser aberto dos lados é que é mais fácil as pessoas conseguirem tocar na eletrónica, daí as SensorBoxes da versão normal serem idealmente instaladas altas na parede. A versão de demonstração será apenas usada em locais protegidos ou controlados como o laboratório de IoT. A base ao ser de madeira facilita a montagem de acessórios que permitam que esta seja agarrada à parede, como se fosse um quadro. Caso se queira colocar em cima de uma mesa podem ser colados autocolantes de feltro ou borracha na parte inferior da base, como se coloca, por exemplo, nos pés das cadeiras.

As SensorBoxes criadas, seguindo os critérios acima descritos, podem ser vistas nas imagens em baixo. Foi usado um transformador AC-DC de 5V e 1.5A com conector macho DC 5.5x2.1x9mm e na SensorBox instalada uma ficha fêmea correspondente. As *breadboards* usadas são as de 400 pinos e antes de as juntar foi retirado a uma delas um dos lados que recebe a corrente negativa e positiva. Os sensores e atuadores externos foram montados com espaçadores de 50mm de alumínio, e os cabos destes foram organizados com pequenos organizadores de cabos pregados na madeira.

4.3.2 Microcontroladores

Para mostrar a compatibilidade da biblioteca criada com vários microcontroladores, foram usados neste trabalho o ESP32, o ESP8266 e o Arduino Uno com placa Wi-Fi.

4.3.2.A ESP32

O ESP32 é o microcontrolador recomendado e escolhido para este trabalho. É usado na SensorBox de demonstração e foi programado usando o FreeRTOS já incluído no seu SDK. Durante o seu uso

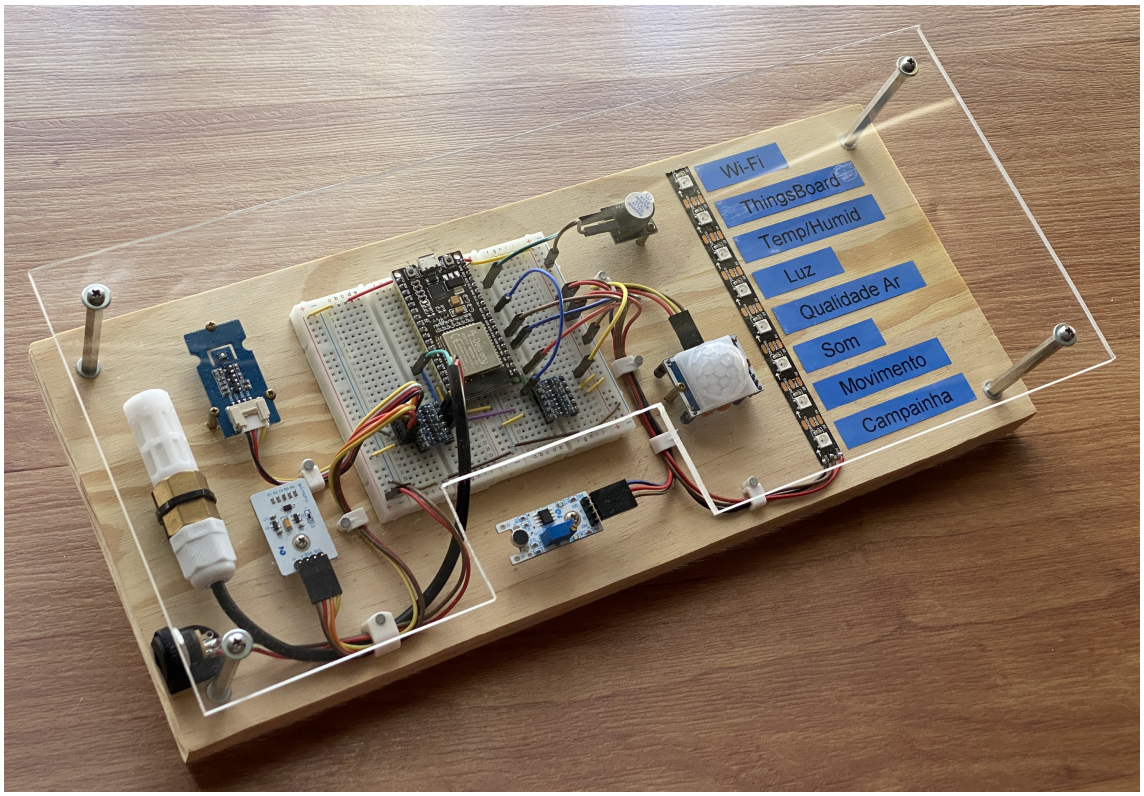


Figura 4.5: SensorBox de demonstração.

percebeu-se que ao utilizar o Wi-Fi a linha dos 3.3V tornava-se um pouco instável, diminuindo assim a estabilidade das leituras analógicas. Foi experimentada a técnica de fazer várias medições sucessivas e usar uma média destas, apesar de melhorar a estabilidade, continuou a não ser uma situação ideal. Podem ser usados filtros que tenham em conta medidas anteriores para melhorar ainda mais a situação, no entanto isto introduz atrasos nos valores face ao valor real. Para resolver este problema pode ser usado um ADC externo para medir os valor analógicos ou, idealmente, usar sensores digitais. Também existe a possibilidade de a instabilidade se dever à versão da placa de desenvolvimento utilizada, no entanto não foi possível testar esta teoria com outra.

4.3.2.B ESP8266

O ESP8266 embora mais velho e pior que o ESP32, continua a ser uma opção bastante viável para este trabalho. É o microcontrolador usado na SensorBox normal e foi utilizado com o seu SDK sem sistema operativo. Usado apenas para testar a compatibilidade com o sistema criado.

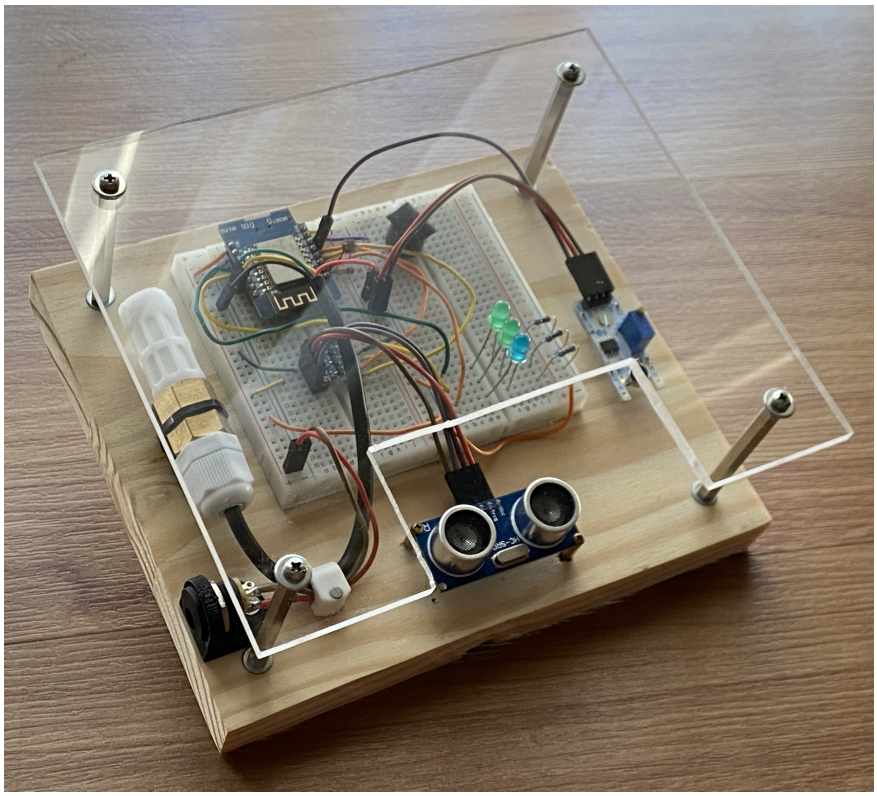


Figura 4.6: SensorBox normal.

4.3.2.C Arduino Uno com placa Wi-Fi

O Arduino Uno é usado para demonstração de compatibilidade pois o IST tem bastantes kits do mesmo e várias placas Wi-Fi. Assim ele pode ser integrado no sistema apesar das suas funcionalidades limitadas, devido à sua reduzida memória e fraco poder de processamento.

No inicio do uso da placa Wi-Fi fornecida pelo IST reparou-se que esta tem um defeito que é conhecido ter acontecido numa série destas placas. Nesta página [31] descrevem como usar a placa e falam sobre a versão com o erro ortográfico ”Moer”, a qual tem um defeito de montagem na parte do conversor lógico que liga a linha de Serial do ESP8266 da placa Wi-Fi ao Arduino Uno. De modo a ultrapassar este problema, por falta de acesso a ferramentas indicadas para corrigir definitivamente o mesmo e por ser uma placa cedida durante o trabalho, optou-se por usar um conversor lógico externo [58](Figura 4.7), assim tanto as linhas do Serial do ESP8266 como as do Arduino Uno são ligadas ao conversor externo. Além disso, os dois pinos da placa Wi-Fi que a ligam ao Serial do Arduino tiveram de ser dobrados de modo a quando a placa é encaixada sobre o Arduino eles não se ligarem, como se pode ver na Figura 4.8. Com esta modificação aproveitamos para usar a placa Wi-Fi através de Serial por *software* [59], podendo usar quaisquer pinos e libertando o Serial do Arduino que é usado para comunicar com o computador, assim é possível imprimir mensagens para o computador enquanto

o código é executado, enquanto antes o Serial estava ocupado a comunicar com a placa Wi-Fi durante execução do código. Neste caso os pinos para o Serial por software definidos na biblioteca são o 12 para o RX e o 13 para o TX.

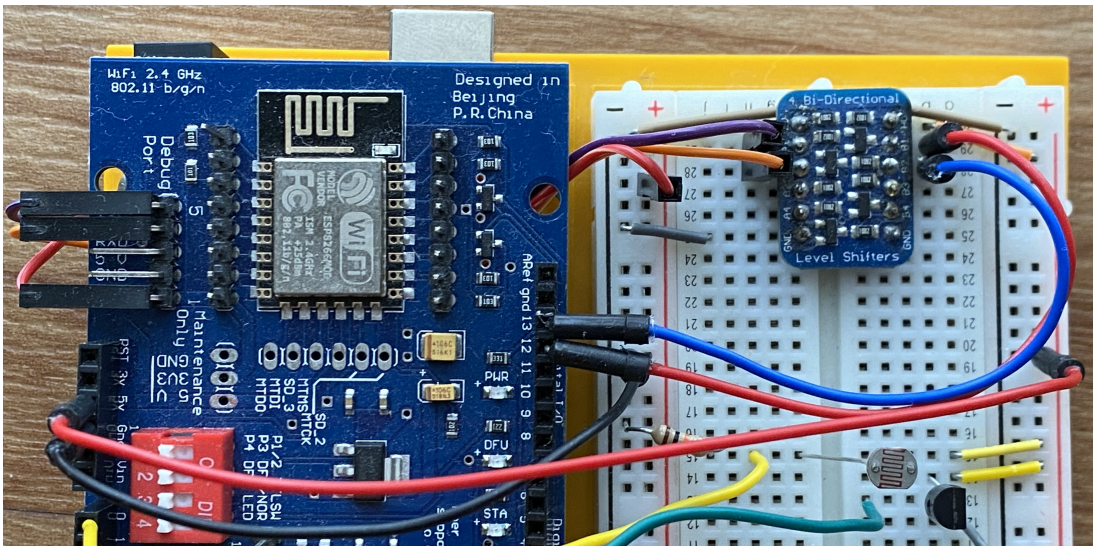


Figura 4.7: Arduino Uno com placa Wi-Fi - conversor lógico externo.

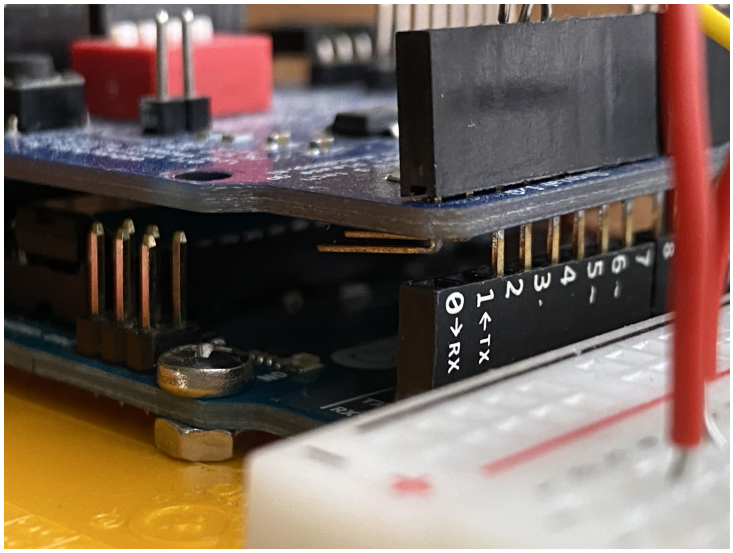


Figura 4.8: Arduino Uno com placa Wi-Fi - pinos do Serial dobrados.

4.3.3 Sensores e Atuadores

Foi criada uma lista de sensores e atuadores interessantes para o laboratório de IoT, pode ser consultada no apêndice B. Os objetivos foram:

- Componentes de baixo custo, tendo em conta que vão ser usados por pessoas sem ou com pouca experiência e que os podem estragar, além de que não serão usados para aplicações profissionais e exigentes, onde a precisão das leituras seja de extrema importância.
- Priorizar sensores digitais, devido à maior estabilidade das leituras efetuadas.
- Sensores principalmente focados em medir o ambiente, o mais útil no caso de uso das SensorBoxes.
- Não repetir sensores já existentes no kit do Arduino Uno oficial, pois o IST já tem vários.
- Sensores populares, de modo a ser mais fácil aos utilizadores encontrar um tutorial que os ajude a conhecer os sensores em causa e a resolver qualquer problema que tenham.

A lista a que se chegou não tem só em conta as SensorBoxes mas também os usos possíveis em trabalhos de laboratório de várias disciplinas. Aplicar atuadores nas SensorBoxes requer ponderação sobre o uso que lhes será dado e a influência destes no ambiente em que vão ser usadas, se vão causar incomodo.

As duas SensorBoxes montadas neste trabalho usam sensores disponíveis no inventário pessoal e no kit oficial do Arduino Uno fornecido pelo IST. Assim alguns sensores usados não são referidos na lista anterior. A versão de demonstração usa os seguintes sensores e atuadores:

- **Temperatura/Humidade:** Usa o sensor SHT30, dentro de um invólucro que o protege de água e pó, e comunica através de I2C. Mais preciso que os sensores da lista mas mais caro e mais difícil de encontrar à venda.
- **Intensidade da luz:** Usa o sensor BH1750FVI [60] e mede a intensidade da luz em lux. Comunica através de I2C.
- **Compostos orgânicos voláteis:** Usa o sensor SGP30, referido na lista, para medir os TVOCs e o eCO2. Comunica através de I2C.
- **Som:** Para medir o ruído é usado o microfone referido na lista, usando a saída digital que acusa quando o ruído sobe acima de um certo limite, este é definido através de um potenciómetro no circuito do microfone.
- **Movimento:** Usa o sensor PIR HC-SR501 referido na lista. Está configurado para acusar quando deteta movimento e só para de acusar quando tiver passado um intervalo de tempo sem mais movimento. Caso detete movimento dentro desse intervalo reinicia a contagem do tempo. O intervalo de tempo é definido no circuito do sensor através de um potenciómetro.

- **Alarme:** Usa uma campainha ativa com a sua própria placa de circuito. Enviando um sinal digital alto (3.3V ou 5V) emite som, enviando o sinal baixo (0V) não emite som.

- **LEDs de estado:** É usada uma fita de oito LEDs RGB individualmente endereçáveis, WS2812B. Sem contar com os fios de alimentação, apenas é preciso um fio para controlar todos os LEDs incluindo as suas cores. Os dois primeiros LEDs são usados para mostrar os estado da ligação Wi-Fi e ao ThingsBoard respetivamente, ficando vermelho se não houver ligação e verde se houver. Os restantes LEDs estão atribuídos cada um a um sensor ou atuador e quando esse envia ou recebe mensagem o seu LED pisca.

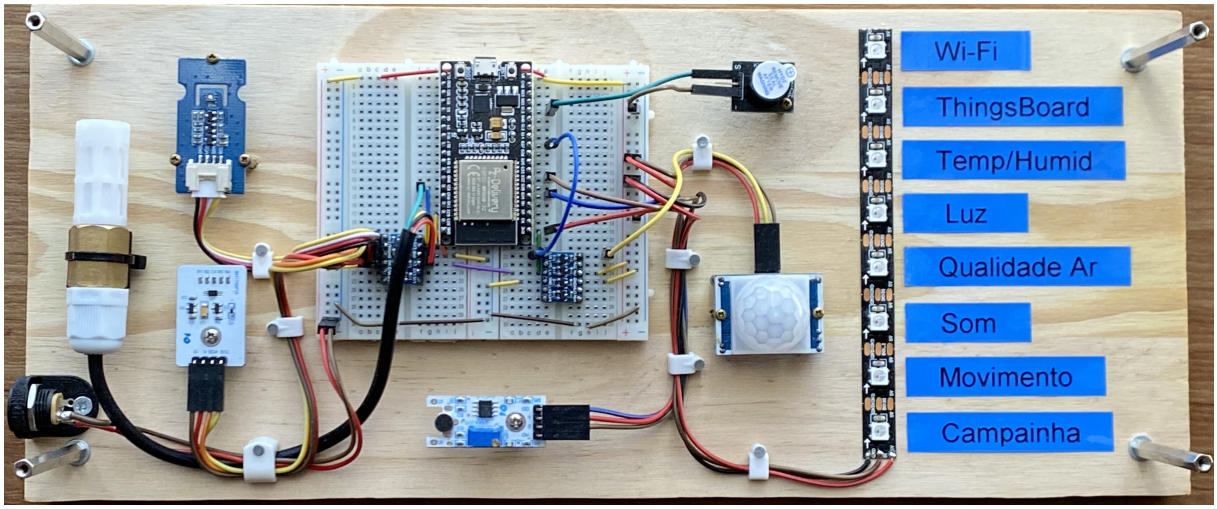


Figura 4.9: SensorBox de demonstração - vista da montagem.

A versão normal da SensorBox usa os seguintes sensores e atuadores:

- **Temperatura/Humidade:** Usa o mesmo sensor SHT30 que a versão de demonstração.

- **Intensidade da luz:** Mede analogicamente a intensidade de luz com recurso a uma fotoresistência. Incluída no kit do Arduino Uno.

- **Som:** É uso o mesmo microfone que na versão de demonstração e está ligado ao microcontrolador da mesma forma.

- **Movimento:** Usa o sensor de ultrassons HC-SR04 referido na lista. Através da distância percebe se houve movimento à frente dele, se a distância for menor que um limite definido.

- **Alarme:** Usa uma campainha ativa como a versão de demonstração, sendo a única diferença que esta se monta diretamente na *breadboard* ao invés de ter a sua própria placa de circuitos. Assim poupa espaço na base de montagem da SensorBox.

- **LEDs de estado:** Usa três LEDs normais de cor única. Dois verdes, sendo um para o estado da ligação Wi-Fi e outro para o estado da ligação ao ThingsBoard. O terceiro LED é azul e pisca quando alguma mensagem, respetiva a sensores ou atuadores, é recebida ou enviada.

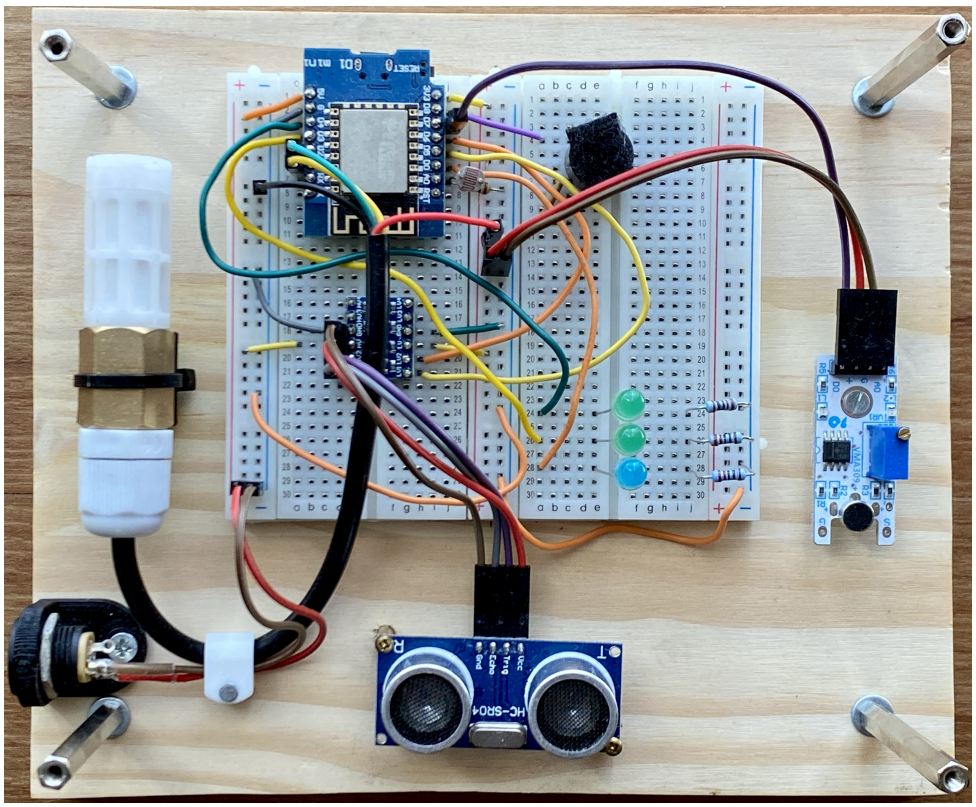


Figura 4.10: SensorBox normal - vista da montagem.

Por fim, foi também montado o kit do Arduino Uno com a placa Wi-Fi e uma *breadboard*. Usa o mesmo sensor de intensidade de luz que a SensorBox normal e o mesmo esquema de LEDs de estado. Usa ainda um sensor de temperatura analógico incluído no seu kit, o TMP36 [61].

4.3.4 Código

Para demonstrar diferentes maneiras de usar a biblioteca criada, a SensorBox Normal, com o ESP8266, foi programada sem recurso a um sistema operativo ou similar [62], e a versão de demonstração, com o ESP32, recorre ao FreeRTOS no seu programa [63]. O Arduino Uno foi programado da mesma forma que o ESP8266 apenas adaptando o código aos seus sensores [64]. O código pode ser encontrado no repositório da biblioteca como exemplos de uso e contem comentários a explicar o que acontece nele.

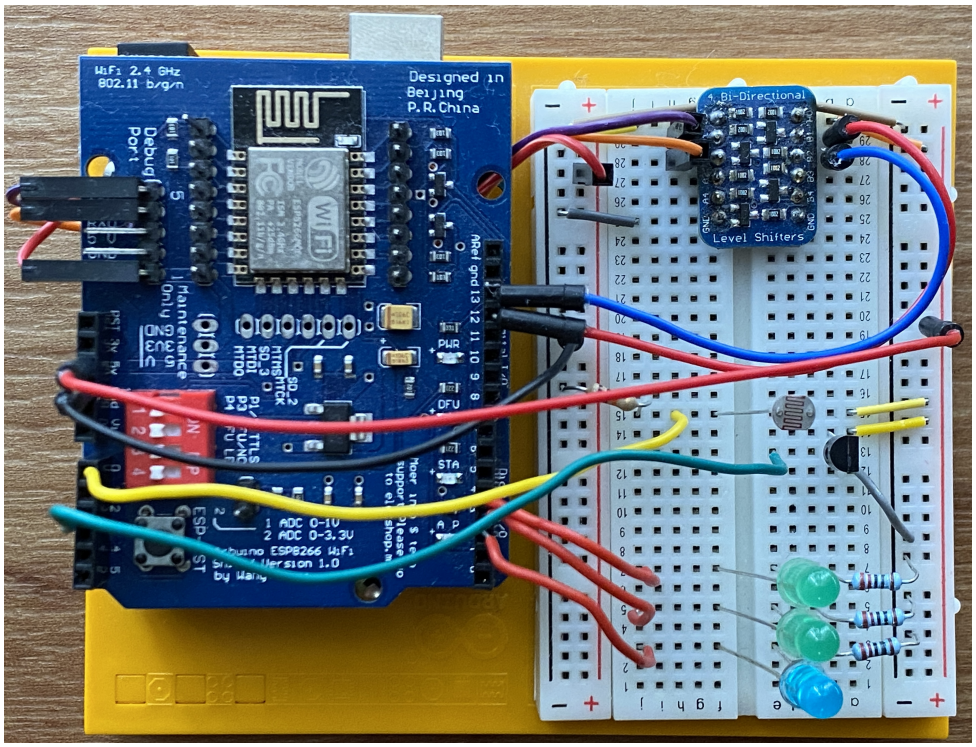


Figura 4.11: Arduino Uno com placa Wi-Fi - vista da montagem.

5

Avaliação

Conteúdo

Neste capitulo são testadas as capacidades do trabalho criado para verificar se cumpre o previsto e se é de facto benéfico para os alunos do IST.

5.1 Desempenho

O desempenho do sistema depende das condições em que é usado, por exemplo, se o servidor e os dispositivos estão na mesma rede ou se comunicam através da internet, ou recursos disponíveis do servidor em que o ThingsBoard está instalado. Os testes realizados usaram para comunicação local o ambiente de produção no servidor local e para comunicação através da internet o ambiente de produção final, ambos referidos na secção 4.1.1.

5.1.1 Latência na telemetria

É importante que os dados dos sensores cheguem ao servidor e a quem os utiliza através da API rapidamente. Numas situações a rapidez das mensagens é mais importante que outras, por exemplo, alguém que esteja subscrito aos dados da temperatura do ar de um sensor não deverá ter problemas que a temperatura esteja 10 segundos desfasada da real pois é uma medição que não varia rapidamente. No entanto, se uma pessoa usar a plataforma com sensores de movimento, ou para gerir interruptores de luz inteligentes, não vai ser agradável para o utilizador do dispositivo ter um atraso de 10 segundos entre ele mandar apagar as luzes e elas realmente se apagarem. Nestas situações o ideal é a latência ser inferior a 1 segundo.

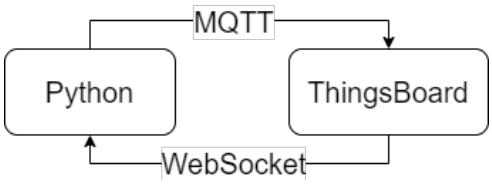


Figura 5.1: Comunicações do teste de latência.

Para medirmos a latência das mensagens será usado um programa em Python que envie telemetria representando um dispositivo e que ao mesmo tempo esteja subscrito através de WebSocket aos dados do seu próprio dispositivo. Logo após ser enviada a mensagem é iniciado um temporizador que acaba quando a mensagem é recebida de volta. A telemetria enviada é um inteiro com o nome ”count”e sai com uma frequência de uma por segundo durante um tempo predefinido. Deste modo conseguimos medir corretamente o tempo que a mensagem passa desde que sai de um dispositivo até ser recebida por outro, passando pelo servidor. Neste teste não estamos só a medir o tempo que a mensagem demora a chegar ao servidor mas sim, chegar ao servidor, ser tratada, enviada para quem está subscrito e recebida por estes. O código de teste pode ser encontrado no repositório da biblioteca para

SensorBoxes, na pasta ”python_test”com o nome ”latency_test.py”.

Os resultados dos testes podem ser vistos na tabela 5.1.

Tabela 5.1: Comparação da latência entre um servidor na rede local e um na internet.

	Servidor Internet	Servidor Local
Tamanho da amostra	500	500
Ping ao servidor	10 ms	$<$ 1 ms
Média	206 ms	9 ms
Mediana	199 ms	9 ms
Desvio Padrão	73 ms	2 ms
Mínimo	92 ms	6 ms
Máximo	395 ms	16 ms
Amplitude	303 ms	10 ms

Como era de esperar, a versão em que o servidor está na mesma rede local que o dispositivo tem menor latência pois as mensagens percorrem um caminho muito menor. Mesmo tendo serviços de fila de mensagens diferentes não deverá praticamente influenciar pois o servidor apenas estava a tratar as poucas mensagens deste dispositivo, mais nenhum estava a comunicar com ele. De notar que os resultados do teste com o servidor a comunicar através da internet eram mais instáveis ao longo das várias vezes que se correu o teste, no entanto os valores colocados na tabela estão dentro do leque dos resultados mais típicos. O resultado do teste com o servidor local era virtualmente sempre o mesmo. Em ambas as situações obtemos valores bastante inferiores a 1 segundo que era o pretendido. Os testes foram realizados em situações ideais de carga para o servidor, apenas um dispositivo comunicava com ele, no entanto isto demonstra que um servidor corretamente dimensionado para o seu caso de uso consegue atingir bons resultados. O seu caso de uso inclui medir variáveis que se vão alterando ao longo do tempo com baixa frequência, ou seja, que não se estão sempre a alterar várias vezes a cada segundo. Também não se espera que seja necessário realizar comunicações exigentes em termos de latência, ou seja, reagir em poucas milésimas de segundo, por exemplo acionar atuadores que necessitem de reagir instantaneamente. Um exemplo que pode incluir casos de uso adequados à plataforma e outros não adequados pode ser um ambiente com robôs que se deslocam, estes não deve depender da plataforma para evitar colidir uns com os outros pois se estes se deslocarem depressa o suficiente o tempo de comunicar a colisão iminente poderá ser grande demais. No entanto, os robôs poderão partilhar estatísticas sobre as suas deslocações para serem visualizadas pelo utilizador ou analisadas mais tarde, como por exemplo, tempo que demorou a alcançar metas, nível da bateria, versão de *firmware*, entre outros.

5.1.2 Tempo de envio de telemetria

Para além da latência de envio das mensagens é importante ver se desde o momento em que se decide enviar uma mensagem até ela ser enviada ocorre algum intervalo de tempo significativo. Para isso foi criado código que envia uma mensagem e mede o tempo que a função de envio demora a executar, pode ser visto no repositório da biblioteca na pasta ”examples”com o nome ”message_sent_test”. Faz isso 100 vezes e no fim devolve as estatísticas sobre as medições, as quais podem ser vistas na tabela 5.2. É feita a comparação dos valores dos três microcontroladores abordados neste trabalho, o ESP32 com o SDK baseado em FreeRTOS, o ESP8266 com o SDK NonOS e o Arduino Uno com placa Wi-Fi. Graças à biblioteca criada, o mesmo código é usado em todos eles.

Tabela 5.2: Comparação entre microcontroladores do tempo de envio de uma mensagem.

	ESP32	ESP8266	Arduino Uno c/ placa Wi-Fi
Tamanho da amostra	100	100	100
Média	1,89 ms	0,94 ms	114 ms
Mediana	2 ms	1 ms	114 ms
Desvio Padrão	0,0 ms	0,46 ms	0,3 ms
Mínimo	1 ms	0 ms	111 ms
Máximo	2 ms	2 ms	156 ms
Amplitude	1 ms	2 ms	45 ms

Ao analisarmos a tabela 5.2 vemos que o tempo demorado pelo ESP32 e o ESP8266 é bastante reduzido, não apresentando um problema. No entanto o Arduino Uno demora bastante mais a enviar a mensagem, isto deve-se ao seu inferior poder de processamento e maioritariamente ao facto de ter de comunicar com a placa Wi-Fi para que seja esta a enviar a mensagem pretendida, tudo isto causa um atraso considerável quando pretendemos que o utilizador tenha a experiência mais próximo do tempo real possível. Assim não se recomenda o uso do Arduino Uno quando ter um atraso mínimo nas medições é importante para o utilizador.

5.1.3 Carga no servidor

Neste teste pretende-se comparar o desempenho das duas configurações referidas no teste da latência. Foi usado o mesmo código que no teste da latência mas agora com mais do que 1 dispositivo a enviar uma mensagem por segundo. Várias combinações de número de dispositivos e número de mensagens por segundo serão usadas. É esperado que o servidor alojado na internet tenha um fraco desempenho devido aos recursos disponíveis e à configuração utilizada, fila de mensagens em memória e base de dados PostgresSQL. O servidor local espera-se que tenha um desempenho bastante superior quando comparando com o anterior por já usar a fila de mensagens Kafka e por o servidor ter mais recursos

disponíveis, ainda assim o desempenho poderia ser melhor utilizando a opção de base de dados híbrida em que para a receção de telemetria era usado a Cassandra em vez de PostgresSQL. De relembrar que este código não testa apenas o envio de mensagens em quantidade para o servidor, mas também a capacidade de o servidor as reencaminhar para quem esteja subscrito a elas. Neste teste cada dispositivo tem um utilizador subscrito às suas mensagens. A duração de cada teste é de 60 segundos. Foram escolhidas três combinações de parâmetros para o teste:

- **Teste 1**: 10 dispositivos a enviar 1 mensagem a cada 2 segundos segundo - um ambiente de pouco uso do servidor, por exemplo, 10 SensorBoxes espalhadas pelo IST com 10 utilizadores subscritos aos seus valores, cada um a um dispositivo. Ritmo de 5 mensagens por segundo, 300 mensagens por minuto.
- **Teste 2**: 50 dispositivos a enviar 1 mensagem por segundo - um ambiente um pouco mais intenso, por exemplo, 10 SensorBoxes espalhadas pelo IST, mais duas turmas de 20 alunos todos com um dispositivo. Todos os 50 a enviar uma mensagem por segundo e com um utilizador subscrito a cada um dos dispositivos. Ritmo de 50 mensagens por segundo, 3.000 mensagens por minuto.
- **Teste 3**: 250 dispositivos a enviar 1 mensagem por segundo - um ambiente de uso já mais intenso em que existem 250 dispositivos a publicar todos os segundos com utilizadores subscritos a essas mensagens. Ritmo de 250 mensagens por segundo, 15.000 mensagens por minuto.

Não se aumentou a frequência de envio de mensagens pois em IoT o típico é os dispositivos enviarem atualizações com pouca frequência, para pouparem energia e porque normalmente as condições não se alteram de forma rápida. Mesmo para sensores que se atualizem frequentemente, um segundo de intervalo já é um valor realista, por exemplo, sensores de velocidade do vento que se atualizem a cada segundo. Os resultados dos testes podem ser vistos na tabela 5.3.

Tabela 5.3: Comparação de carga entre um servidor na rede local e um na internet.

	Servidor Internet			Servidor Local		
Teste	1	2	3	1	2	3
Média	216 ms	698 ms	1141 ms	85 ms	115 ms	500 ms
Mediana	211 ms	682 ms	1151 ms	85 ms	109 ms	448 ms
Desvio Padrão	68 ms	169 ms	470 ms	23 ms	58 ms	203 ms
Mínimo	95 ms	332 ms	134 ms	22 ms	6 ms	174 ms
Máximo	388 ms	1248 ms	2774 ms	157 ms	449 ms	1055 ms
Amplitude	293 ms	916 ms	2640 ms	137 ms	443 ms	881 ms
Mensagens Falhadas	0	0	46	0	0	0

Tal como ocorreu no teste da latência, os valores dos testes do servidor na internet eram mais instáveis, no entanto os valores selecionados representam a maioria dos resultados obtidos. Como

esperado o servidor local teve um desempenho bastante superior, mantendo-se praticamente sempre a baixo dos 1 segundos de latência, o máximo desejado. O servidor na internet devido ao caminho mais longo que as mensagens percorrem e à configuração do servidor, no teste 2 já tinha pior desempenho que o servidor local no teste 3. No terceiro teste a latência já está maioritariamente sempre a cima de um segundo e já são perdidas mensagens. Assim o servidor alojado na RNL é apenas indicado para desenvolvimento e cargas muito ligeiras, para um uso normal, dentro do esperado tendo em conta este trabalho, o servidor deveria ser pedido com mais recursos de RAM e consequentemente a possibilidade de instalar a fila de mensagens Kafka nele. Em ambos também seria benéfico o uso da base de dados híbrida, sendo Cassandra usada assim para a receção de mensagens.

Existem testes oficiais de desempenho do ThingsBoard nos servidores da AWS [65]. Os testes não são diretamente comparáveis com os aqui realizados pois os oficiais apenas testam a publicação de mensagens [66], não a subscrição às mesmas por utilizadores. No entanto ambos os tipos de testes complementam-se e dão uma ideia geral do desempenho dos servidores no envio e subscrição de mensagens e os seus atrasos.

5.2 Tamanho da biblioteca

Foi medido o impacto da biblioteca oficial e da criada neste trabalho na memória dos microcontroladores. As bibliotecas quando estão a ser compiladas descartam o código que não interessa para o microcontrolador em questão. O Arduino Uno tem as suas funcionalidades limitadas, apenas consegue enviar informação, não consegue subscrever, assim quando as bibliotecas são compiladas não incluem o código relativo a subscrições. O ESP32 tem uma funcionalidade extra face ao ESP8266, inclui a capacidade de se conectar a redes Wi-Fi com autenticação WPA2 Enterprise. Estas diferenças entre funcionalidades e o facto de os microcontroladores usarem bibliotecas diferentes para aceder ao Wi-Fi e à EEPROM faz com que, no final da compilação, o tamanho da biblioteca varie bastante entre eles. Na tabela 5.4 estão os resultados de compilar um ficheiro de código vazio, a compilação do ficheiro vazio apenas incluindo a biblioteca oficial do ThingsBoard, que por sua vez já inclui outras bibliotecas, e por fim o ficheiro vazio incluindo apenas a biblioteca SensorBox criada neste trabalho, que por sua vez inclui a oficial do ThingsBoard e outras necessárias às funcionalidades que acrescenta. Os valores da tabela quando são compiladas as bibliotecas são sempre comparados à compilação doo ficheiro vazio, mostra quantos bytes a mais foram necessários, assim é mais fácil de interpretar o gráfico face a ter sempre os totais. É mostrado a ocupação do armazenamento do microcontrolador e também a ocupação da memória dinâmica, esta última é onde o código cria e manipula variáveis enquanto corre.

Os níveis de ocupação entre as duas bibliotecas diferem consideravelmente, no entanto, na biblioteca da SensorBox existe código incluído que será sempre necessário criar para usar a biblioteca

Tabela 5.4: Comparação da ocupação de memória da biblioteca SensorBox.

	ESP32		ESP8266		Arduino Uno c/ placa Wi-Fi	
	Armazenamento	M. dinâmica	Armazenamento	M. dinâmica	Armazenamento	M. dinâmica
Vazio	197.736 bytes	13.084 bytes	256.688 bytes	26.776 bytes	444 bytes	9 bytes
ThingsBoard	+57.170 bytes	+636 bytes	+3.684 bytes	+116 bytes	+1.970 bytes	+221 bytes
SensorBox	+75.958 bytes	+892 bytes	+12.452 bytes	+2.352 bytes	+4.420 bytes	+512 bytes

oficial, por exemplo, não é possível utilizar a biblioteca oficial sem o microcontrolador estar ligado à internet, assim o utilizador vai ter sempre de importar a biblioteca do Wi-Fi e criar o código para se conectar e manter a ligação, o que irá reduzir a diferença de ocupação. Tendo em conta que o ESP32 e o ESP8266 têm bastante memória, a importação da biblioteca SensorBox não será um problema mesmo quando não são usadas todas as funcionalidades, traz sempre a vantagem de ficar com um código mais organizado pois muito dele passa a estar já incluído na biblioteca. Para o Arduino Uno a memória já é um recurso mais escasso, no entanto a biblioteca SensorBox já exclui as funcionalidades não compatíveis e assim a maior parte do código que acrescente já vai ter de ser criado manualmente caso não seja utilizada, como por exemplo, a ligação à rede Wi-Fi e ao servidor do ThingsBoard. Assim incluir a biblioteca criada neste trabalho continua a ser uma mais valia mesmo no Arduino Uno.

5.3 Trabalhos de laboratório

Foram criados três guiões de laboratório com base nos da cadeira de ACIC para serem testados com alunos e assim para provar a usabilidade da plataforma num ambiente de aprendizagem. Devido à situação pandémica em que decorreu este trabalho e o facto de os trabalhos de laboratório precisarem de presença física para fornecer o material necessário à sua realização, não foi possível realizar os testes. No entanto, os guiões em que estes se baseiam já foram testados vários anos com alunos e portanto mostram-se adequados. Mantendo a mesma ideia dos guiões de ACIC, apenas os adequando a usar as tecnologias relacionadas com este trabalho sem que a dificuldade aumente, mostra que o sistema aqui criado é adequado ao uso por parte de alunos nas aulas e consequentemente nos projetos. Os guiões podem ser encontrados no repositório da biblioteca das SensorBoxes [46] dentro da pasta ”guides”.

A diferença entre os dois primeiros laboratórios de ACIC e os guiões aqui criados é o uso do ESP32 em vez do Arduino Uno, deste modo os guiões são praticamente os mesmos apenas com as referências a conteúdo relativo aos microcontroladores diferentes, como esquemas de ligação e referências a páginas da internet.

O terceiro laboratório é onde se usa o sistema criado neste trabalho. O objetivo mantém-se face ao original, conectar dois microcontroladores tendo um os atuadores e outro os sensores, o primeiro ao

receber os valores dos sensores aciona os seus atuadores se necessário. Em ACIC a comunicação era feita com fios através de comunicação I2C, no entanto no ano da pandemia, estando os alunos em casa e sendo os trabalhos de grupo, já não foi possível interligar os Arduinos entre si com fios, assim o guião mudou, dando a liberdade aos alunos de, usando o Arduino ligado a um computador, os fazer comunicar através da internet. Com o sistema aqui criado os microcontroladores têm a autonomia de se ligarem sozinhos à internet e consequentemente à uma plataforma de IoT. No guião desenvolvido optou-se por não programar a lógica do guião de laboratório diretamente nos microcontroladores, estes devem ser apenas para enviar dados de sensores e receber comandos para atuadores. A lógica foi implementada numa aplicação Python que recorre à API do ThingsBoard para receber os dados dos sensores e decidir se deve alterar os estados dos atuadores.

Com estes novos guiões, além de ser possível os alunos trabalharem em grupo à distância, também ganham experiência com as plataformas de IoT, importante no mundo profissional onde cada vez mais estas são usadas para ligar os vários objetos existentes na nossa vida. Além disso também ficariam a conhecer a plataforma para usarem em outros trabalhos caso quisessem, como para aceder às SensorBoxes ou adicionar novos dispositivos.

Face ao I2C, a limitação que deve ser considerada é a demora das comunicações como visto nos pontos anteriores desta secção. No entanto, no mundo real e para a utilização indicada no guião, a latência que se está a adicionar às comunicações, usando o ESP32, não é um problema.

6

Conclusão

Conteúdo

Neste capítulo são extraídas as conclusões do trabalho realizado, o que melhorar no futuro e ideias para continuação do mesmo.

6.1 Conclusões

Com este trabalho iniciou-se o desenvolvimento de um laboratório de IoT para o IST, tendo-se optado por começar com a implementação de uma infraestrutura para interligar dispositivos através da internet ao qual os alunos podem ter acesso em qualquer parte do mundo. Tendo como opção apenas plataformas de IoT open-source decidiu-se usar o ThingsBoard devido à sua bastante superior popularidade face às opções concorrentes, o desenvolvimento bastante ativo e um superior leque de funcionalidades sempre a ser aumentado. Foram comparadas 3 populares opções de dispositivos nesta área para quem está a começar, o ESP32, o ESP8266 e o Arduino Uno com placa Wi-Fi. Chegou-se à conclusão que o ESP32 é a melhor opção, sendo o que tem melhor desempenho, melhor conexão ao Wi-Fi, possui mais pinos para se utilizar e custa praticamente o mesmo que as outras opções, no entanto se o Arduino Uno for a versão oficial torna-se bastante mais caro. O Arduino Uno não é ideal para uso em conjunto com o ThingsBoard devido aos seus poucos recursos, além de que a placa Wi-Fi disponibilizada acrescentava complexidade ao seu uso devido aos seus problemas com o conversor de nível lógico. Foram criadas montagens para servir de base à instalação de sensores espalhados no IST e para demonstração do seu funcionamento aos alunos. Foi criada também uma biblioteca para os dispositivos mencionados, sobre a biblioteca oficial, para simplificar ainda mais a comunicação destes com o servidor ao ponto de o mesmo código ser possível de utilizar em todos eles, quando as funcionalidades utilizadas estão disponíveis em todos. Graças à biblioteca foi possível criar um guião de laboratório simples de seguir para realização em aula, sem que os alunos tenham experiência em realizar comunicações através da internet com estes dispositivos.

Através dos testes de desempenho realizados observou-se que um servidor de ThingsBoard corretamente dimensionado é capaz de servir o objetivo do trabalho, capacitar o IST com uma plataforma de IoT utilizável pelos alunos dentro e fora do mesmo para a exploração e a realização de diversos tipos de trabalhos na área, desde os mais relaxados até aos mais exigentes em termos de tempos de resposta, dentro do esperado na área de IoT.

Graças à infraestrutura criada os alunos de vários cursos podem experimentar a Internet das Coisas nas suas áreas de interesse, ganhando assim uma boa base de conhecimentos para o mundo profissional do presente e do futuro onde cada vez mais objetos estão conectados a plataformas de IoT enviando dados sobre o ambiente que os rodeia ou o que eles fazem para mais tarde serem analisados com variados objetivos.

6.2 Trabalho Futuro

Futuros desenvolvimentos deste trabalho passariam pela instalação de várias SensorBoxes no IST, por exemplo, com sensores de medição do estado e da qualidade do ambiente, deste modo os alunos interessados podem realizar projetos sobre detetar a ocupação dos espaços interiores. Outras hipóteses seriam a criação de tutoriais sobre a utilização de outras funcionalidades existentes no ThingsBoard, a criação de uma página na internet para o laboratório de IoT de modo a que os alunos possam aprender sobre o que lhes é disponibilizado tanto em termos de sistemas como de equipamentos, e que possam consultar informação sobre como utilizar estes. Quanto às SensorBoxes, o objetivo futuro seria desenvolver montagens para exterior, em que os componentes estariam protegidos do clima, podendo até ser criado versões alimentadas a energia solar para flexibilidade de instalação no campus sem ter de passar fios elétricos até estas. Por fim, no ambiente de laboratório poderiam ser criadas bancadas de trabalho especializadas para IoT e microcontroladores, onde existiriam placas com várias combinações de sensores e atuadores pré-montados em conjunto com *breadboards* e fontes de alimentação necessárias para os alunos conseguirem mais facilmente e com maior organização experimentar o uso de diversos componentes sem terem de estar a lidar com a montagem destes individualmente e possivelmente os danificando, apenas teriam de ligar os fios dos sensores desejados na montagem que estivessem a realizar.

Bibliografia

[1] Disciplina Aplicações e Computação para a Internet das Coisas. Último acesso a 9-Set-2021. [Online]. Available: https://fenix.tecnico.ulisboa.pt/cursos/meic-a/disciplina-curricular/283003985068088

[2] Disciplina Design de Interação para a Internet das Coisas. Último acesso a 9-Set-2021. [Online]. Available: https://fenix.tecnico.ulisboa.pt/cursos/meic-a/disciplina-curricular/1127428915200128

[3] Disciplina Ambientes Inteligentes. Último acesso a 9-Set-2021. [Online]. Available: https://fenix.tecnico.ulisboa.pt/cursos/meic-a/disciplina-curricular/283003985068090

[4] Disciplina Computação Móvel e Ubíqua. Último acesso a 9-Set-2021. [Online]. Available: https://fenix.tecnico.ulisboa.pt/cursos/meic-a/disciplina-curricular/283003985068054

[5] R. C. Y. Lam, A. Junus, W. M. Y. Cheng, X. Li, and L. C. H. Lam, "Iot application in construction and civil engineering works," in *2017 International Conference on Computational Science and Computational Intelligence (CSCI)*, 2017, pp. 1320–1325.

[6] R. Chacón, H. Posada, Álvaro Toledo, and M. Gouveia, "Development of iot applications in civil engineering classrooms using mobile devices," *Computer Applications in Engineering Education*, vol. 26, pp. 1769 – 1781, 2018. [Online]. Available: https://www.semanticscholar.org/paper/Development-of-IoT-applications-in-civil-classrooms-Chac%C3%B3n-Posada/623a088cdb1f08b7543c2b50255b2332cf19de93

[7] *IoT BASED EDGE AND CLOUD COMPUTING FOR SMART ENVIRONMENTAL ENGINEERING APPLICATIONS*, 2018. [Online]. Available: https://www.researchgate.net/publication/326263861_IoT-based_Edge_and_Cloud_Computing_for_Smart_Environmental_Engineering_Applications

[8] M. Ghobakhloo, "Industry 4.0, digitization, and opportunities for sustainability," *Journal of Cleaner Production*, vol. 252, p. 119869, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0959652619347390

[9] H. N. A. Ibraheem M. Khalil, “Monitoring of water purification process based on iot,” *IOSR Journal of Electronics and Communication Engineering (IOSR-JECE)*, vol. 14, pp. 56–62, Mar-Abr 2019. [Online]. Available: https://www.researchgate.net/profile/Hazem-abdulrazzak/publication/332472343_Monitoring_of_Water_Purification_Process_Based_on_IoT/links/5cb799ad299bf120976ccbbc/Monitoring-of-Water-Purification-Process-Based-on-IoT.pdf

[10] Lab de IoT do ISCTE. Último acesso a 21-Out-2021. [Online]. Available: https://istar.iscte-iul.pt/portfolio-posts/vr-lab/

[11] Lab de IoT de Swami Keshvanand Institute of Technology. Último acesso a 21-Out-2021. [Online]. Available: https://www.skit.ac.in/research/iot-lab.html

[12] Lab de IoT de Rhine-Waal University of Applied Sciences. Último acesso a 21-Out-2021. [Online]. Available: https://www.hochschule-rhein-waal.de/en/faculties/communication-and-environment/laboratories/iot-lab

[13] Lab IoT da universidade de Chicago. Último acesso a 21-Out-2021. [Online]. Available: https://cdac.uchicago.edu/iot-lab/

[14] Lab IoT do politécnico de Milano. Último acesso a 21-Out-2021. [Online]. Available: https://www.polimi.it/en/scientific-research/research-at-the-politecnico/laboratories/interdepartmental-laboratories/internet-of-things-lab/

[15] Lab IoT da universidade de Tartu. Último acesso a 21-Out-2021. [Online]. Available: https://www.cs.ut.ee/en/research/internet-of-things-lab

[16] Rede LoRaWAN. Último acesso a 26-Out-2021. [Online]. Available: https://lora-alliance.org/about-lorawan/

[17] Lab IoT da universidade de Swinburne. Último acesso a 22-Out-2021. [Online]. Available: https://www.swinburne.edu.au/research/facilities-equipment/digital-research-innovation-capability-platform/the-internet-of-things-lab/

[18] Lab IoT da universidade de Sungkyunkwan. Último acesso a 22-Out-2021. [Online]. Available: http://iotlab.skku.edu/

[19] Plataforma de backend para aplicações, Kuzzle. Último acesso a 21-Out-2021. [Online]. Available: https://kuzzle.io/kuzzle-backend/

[20] Página dos serviços da aws iot. Último acesso a 15-Set-2021. [Online]. Available: https://aws.amazon.com/pt/iot/

[21] Página do google iot. Último acesso a 15-Set-2021. [Online]. Available: https://cloud.google.com/solutions/iot

[22] Página do arduino iot cloud. Último acesso a 15-Set-2021. [Online]. Available: https://store.arduino.cc/digital/create

[23] Plataforma de IoT Open-Source - ThingsBoard. Último acesso a 15-Set-2021. [Online]. Available: https://thingsboard.io

[24] Plataforma de IoT Open-Source - Kaa IoT. Último acesso a 15-Set-2021. [Online]. Available: https://www.kaaiot.com/

[25] Plataforma de IoT Open-Source - OpenRemote. Último acesso a 15-Set-2021. [Online]. Available: https://openremote.io/

[26] Plataforma de IoT Open-Source - SiteWhere. Último acesso a 15-Set-2021. [Online]. Available: https://sitewhere.io/en/

[27] L. A. Otalora, “Implementación de una plataforma colaborativa del internet de las coisas para la captura de variables ambientales para el municipio santiago de cali,” Mai 2019, pp. 55–58.

[28] W. P. d. A. Eduardo Natan Bitencourt, “Iot centralization and management applying thingsboard platform,” Aug 2018, pp. 25–27.

[29] Página do Arduino Uno. Último acesso a 9-Set-2021. [Online]. Available: https://store.arduino.cc/arduino-uno-rev3

[30] Website do Arduino. Último acesso a 15-Set-2021. [Online]. Available: https://www.arduino.cc/

[31] Página do shield Wi-Fi. Último acesso a 9-Set-2021. [Online]. Available: https://www.instructables.com/ESP8266-ESP-12E-UART-Wireless-WIFI-Shield-TTL-Conv/

[32] Microcontrolador com Wi-Fi - ESP8266. Último acesso a 15-Set-2021. [Online]. Available: https://www.espressif.com/en/products/socs/esp8266

[33] Website da Espressif. Último acesso a 15-Set-2021. [Online]. Available: https://www.espressif.com/en

[34] ESP8266 Wemos D1 Mini. Último acesso a 15-Set-2021. [Online]. Available: https://www.wemos.cc/en/latest/d1/d1_mini.html

[35] Arduino IDE. Último acesso a 15-Set-2021. [Online]. Available: https://www.arduino.cc/en/software

[36] Microcontrolador com Wi-Fi e Bluetooth - ESP32. Último acesso a 15-Set-2021. [Online]. Available: https://www.espressif.com/en/products/socs/esp32

[37] Sistema Operativo em tempo real para microcontroladores - FreeRTOS. Último acesso a 9-Set-2021. [Online]. Available: https://www.freertos.org

[38] FreeRTOS para Arduino. Último acesso a 11-Out-2021. [Online]. Available: https://github.com/feilipu/Arduino_FreeRTOS_Library

[39] Website do Docker. Último acesso a 9-Set-2021. [Online]. Available: https://www.docker.com/

[40] Website do Ubuntu. Último acesso a 16-Set-2021. [Online]. Available: https://ubuntu.com/

[41] Página Eduroam. Último acesso a 16-Out-2021. [Online]. Available: https://eduroam.pt/

[42] ThingsBoard - Instalar com o Docker no Windows. Último acesso a 1-Out-2021. [Online]. Available: https://thingsboard.io/docs/user-guide/install/docker-windows/

[43] ThingsBoard - Instalar com o Docker no MacOS ou Linux. Último acesso a 1-Out-2021. [Online]. Available: https://thingsboard.io/docs/user-guide/install/docker/

[44] ThingsBoard - Instalar no Ubuntu Server. Último acesso a 1-Out-2021. [Online]. Available: https://thingsboard.io/docs/user-guide/install/ubuntu/

[45] ThingsBoard - Portos a abrir para o uso do servidor. Último acesso a 1-Out-2021. [Online]. Available: https://thingsboard.io/docs/user-guide/install/cluster/aws-self-hosted-setup/

[46] Biblioteca para Arduino, SensorBox-ThingsBoard-SDK. Último acesso a 28-Set-2021. [Online]. Available: https://github.com/JoaoSaramago/SensorBox-ThingsBoard-SDK

[47] ThingsBoard - REST API. Último acesso a 5-Out-2021. [Online]. Available: https://demo.thingsboard.io/swagger-ui.html

[48] ThingsBoard - Opções de uso da REST API. Último acesso a 5-Out-2021. [Online]. Available: https://thingsboard.io/docs/api/

[49] ThingsBoard - Biblioteca em Python para acesso à REST API. Último acesso a 5-Out-2021. [Online]. Available: https://thingsboard.io/docs/reference/python-rest-client/

[50] ThingsBoard - versão da biblioteca utilizada. Último acesso a 15-Set-2021. [Online]. Available: https://github.com/thingsboard/thingsboard-arduino-sdk/tree/23aaf47f4838e887493aeafaef92f76e4159bfb7

[51] Repositório SDK NonOS 3.0.4 para placa Wi-Fi. Último acesso a 27-Set-2021. [Online]. Available: https://github.com/espressif/ESP8266_NONOS_SDK/releases/tag/v3.0.4

[52] Instalar o ESP32 no Arduino IDE. Último acesso a 28-Set-2021. [Online]. Available: https://randomnerdtutorials.com/installing-the-esp32-board-in-arduino-ide-windows-instructions/

[53] Instalar o ESP8266 no Arduino IDE. Último acesso a 28-Set-2021. [Online]. Available: https://randomnerdtutorials.com/how-to-install-esp8266-board-arduino-ide/

[54] Problema com ESP8266 no Arduino IDE e o MacOS Big Sur. Último acesso a 28-Set-2021. [Online]. Available: https://github.com/esp8266/Arduino/issues/7763

[55] Instalar bibliotecas no Arduino IDE. Último acesso a 28-Set-2021. [Online]. Available: https://www.arduino.cc/en/guide/libraries

[56] ESP32 Eduroam. Último acesso a 30-Nov-2021. [Online]. Available: https://github.com/martinius96/ESP32-eduroam

[57] Kit iniciante oficial do Arduino Uno. Último acesso a 16-Out-2021. [Online]. Available: https://store.arduino.cc/collections/kits/products/arduino-starter-kit-multi-language

[58] Conversores de sinais lógicos. Último acesso a 16-Out-2021. [Online]. Available: https://www.ptrobotics.com/blog/post/conversor-bidirecional-33v-5v.html

[59] Página sobre Serial por software. Último acesso a 16-Out-2021. [Online]. Available: https://www.arduino.cc/en/Reference/softwareSerial

[60] Sensor de intensidade de luz BH1750. Último acesso a 16-Out-2021. [Online]. Available: https://www.velleman.eu/products/view/?id=450560

[61] Sensor de temperatura analógico TMP36. Último acesso a 16-Out-2021. [Online]. Available: https://learn.adafruit.com/tmp36-temperature-sensor

[62] Código da SensorBox Normal com o ESP8266. Último acesso a 6-Out-2021. [Online]. Available: https://github.com/JoaoSaramago/SensorBox-ThingsBoard-SDK/blob/main/examples/esp8266-normal/esp8266-normal.ino

[63] Código da SensorBox Demonstração com o ESP32. Último acesso a 6-Out-2021. [Online]. Available: https://github.com/JoaoSaramago/SensorBox-ThingsBoard-SDK/blob/main/examples/esp32-demonstracao-free_rtos/esp8266-esp32-demonstracao-free_rtos.ino

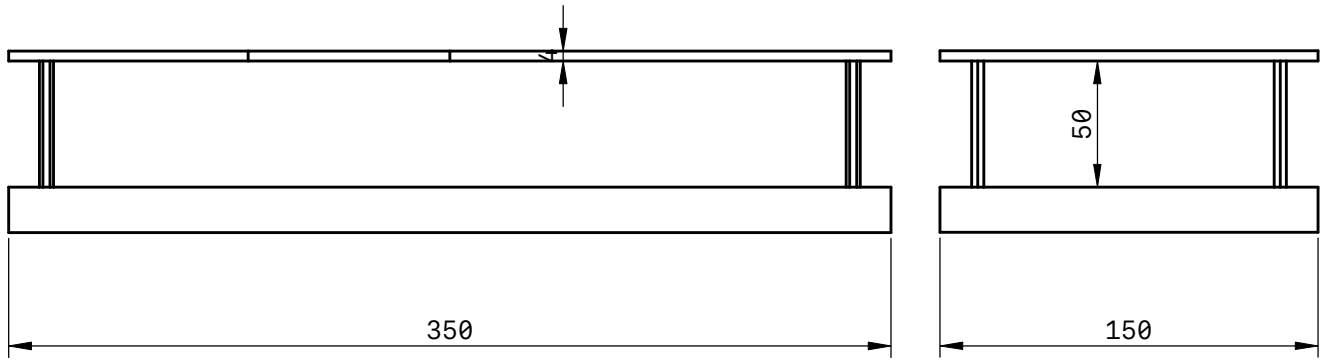
[64] Código do Arduino Uno. Último acesso a 6-Out-2021. [Online]. Available: https://github.com/JoaoSaramago/SensorBox-ThingsBoard-SDK/blob/main/examples/arduino-uno/arduino-uno.ino

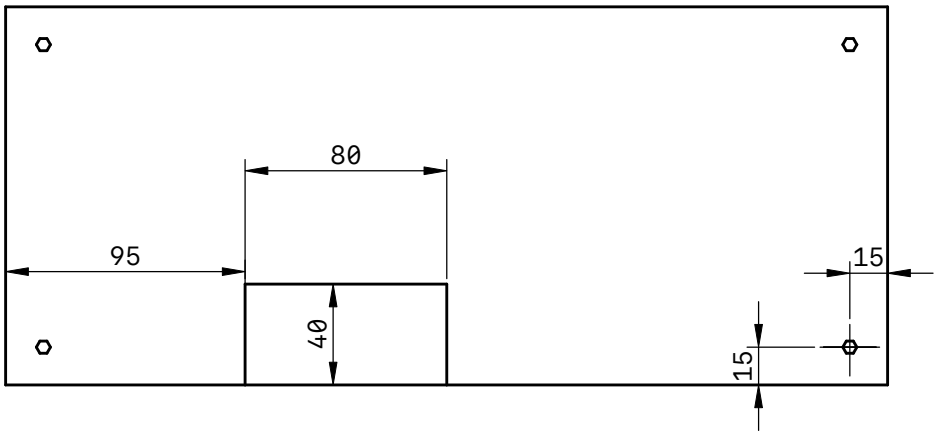
[65] Testes de desempenho do ThingsBoard em diferentes instâncias do AWS. Último acesso a 15-Out-2021. [Online]. Available: https://thingsboard.io/docs/reference/performance-aws-instances/

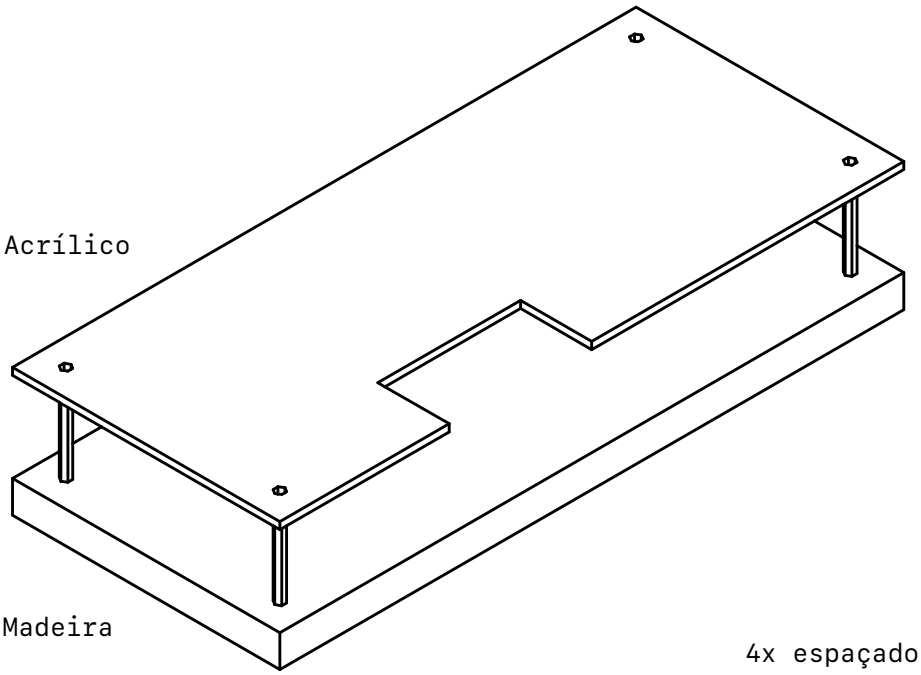
[66] Testes de desempenho do ThingsBoard - Ferramentas de teste. Último acesso a 15-Out-2021. [Online]. Available: https://thingsboard.io/docs/reference/performance-tools/#performance-tests-project

A

Esquema das SensorBoxes





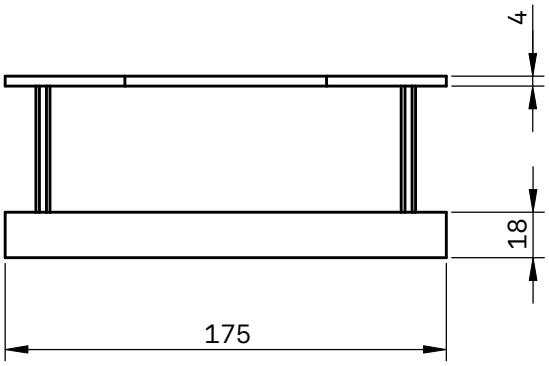


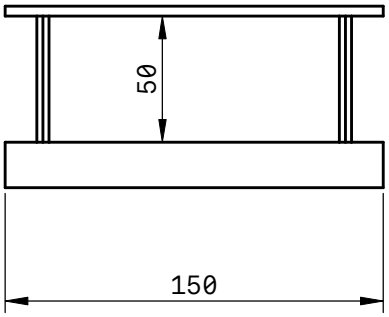
Acrílico

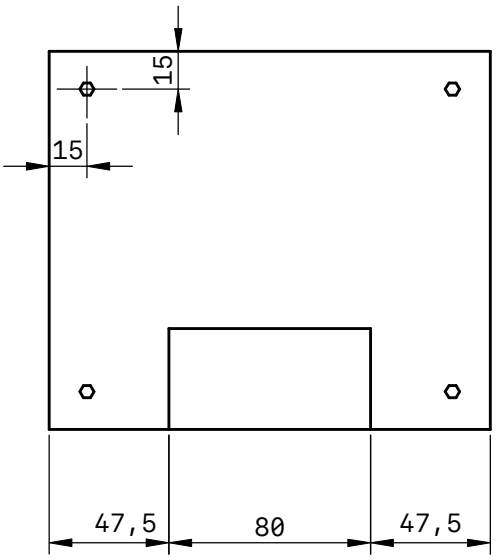
Madeira

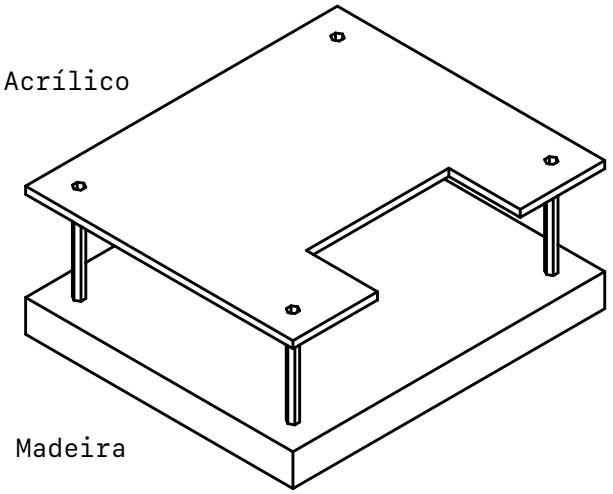
4x espaçador M3 50mm

TÍTULO SensorBox Demonstração		
ÚLTIMA ATUALIZAÇÃO 09/21/21	UNIDADES mm	FOLHA 1 / 1
PROJEÇÃO DE PRIMEIRO ÂNGULO	ESCALA 1:3	TAMANHO A4









4x espaçador M3 50mm

TÍTULO SensorBox Normal		
ÚLTIMA ATUALIZAÇÃO 09/21/21	UNIDADES mm	FOLHA 1 / 1
PROJEÇÃO DE PRIMEIRO ÂNGULO	ESCALA 1:3	TAMANHO A4

B

Lista de Sensores e Atuadores

Última vez atualizada a 24 de setembro de 2021.

Sensores

Tipo	Nome	Descrição	Link loja Portugal	Preço	Link especificações
Temperatura e humidade	Funduino DHT11	Temperatura 0º a 50ºC (precisão 2ºC), humidade 20-80% (precisão 5%)	https://mauser.pt/catalog/product_info.php?cPath=1667_2669_2676&products_id=096-7807	3,00€ (1 uni)/ 1,98€ (5+ uni)	https://funduino.de/anleitung-dht11-dht22
Temperatura e humidade	Funduino DHT22	Temperatura -40º a 125ºC (precisão 0.5ºC), humidade 0-100% (precisão 2-5%)	https://www.ptrobotics.com/atmosfericos/5111-funduino-dht22-temperature-and-humidity-sensor.html	9,72€	https://funduino.de/anleitung-dht11-dht22
Temperatura	DS18B20	Temperatura -55º a 125ºC (precisão 0.5ºC)	https://www.ptrobotics.com/sensores-de-temperatura/1307-ds18b20-one-wire-digital-temperature-sensor.html	3,20€	https://cdn.sparkfun.com/datasheets/Sensors/Temp/DS18B20.pdf
Acelerômetro/Giroscópio	MPU-6050	Usa I2C	https://www.ptrobotics.com/acelerometros/3639-triple-axis-accelerometer-gyro-mpu-6050.html	6,09€	https://dlnmh9ip6v2uc.cloudfront.net/datasheets/Components/General%20IC/PS-MPU-6000A.pdf
Bússola/Acelerômetro	GY-511	Usa I2C	https://www.ptrobotics.com/acelerometros/7207-gy-511-lsm303dlhc-compass-3-axis-accelerometer-and-magnetometer-sensor-module.html	9,59€	https://cdn-shop.adafruit.com/datasheets/LSM303DLHC.PDF
Bússola/Acelerômetro/Giroscópio	MPU-9250	Pode usar SPI ou I2C	https://www.ptrobotics.com/imu/8849-giroscopio-de-9-eixos-acelerometro-e-bussola-com-spi-e-i2c-mpu-9250.html	8,61€	https://invensense.tdk.com/wp-content/uploads/2015/02/PS-MPU-9250A-01-v1.1.pdf
Medido pulsação		Sensor analógico	https://www.ptrobotics.com/biometrica/5908-pulse-heart-rate-sensor-module.html	8,12€	
Med. pulsação e oxigénio	MAX30102	Usa I2C	https://abcrobotica.pt/produto/sensor-de-frequencia-cardiaca-max30102-keyestudio/	10,48€	https://datasheets.maximintegrated.com/en/ds/MAX30102.pdf
Ultrasom	HC-SR04	Usa 2 pinos	https://mauser.pt/catalog/product_info.php?cPath=1667_2669_2677&products_id=096-6220	2,2€ (1 uni) / 1,48€ (10+ uni)	http://web.eece.maine.edu/~zhu/book/lab/HC-SR04%20User%20Manual.pdf
Ultrasom	HY-SRF05	Usa 1 ou 2 pinos	https://www.ptrobotics.com/sensor-ultrasom/6065-hy-srf05-ultrasonic-distance-sensor-module.html	4,80€	https://www.robot-electronics.co.uk/htm/srf05tech.htm
Movimento (PIR)	HC-SR501	Sinal HIGH a 3.3V mas funciona com o Arduino	https://mauser.pt/catalog/product_info.php?cPath=1667_2669_2675&products_id=096-8205	2,4€ (1 uni) / 1,89€ (5+ uni)	https://www.epitran.it/ebayDrive/datasheet/44.pdf
Qualidade do Ar	MQ-135	Detecção de Gás Amónia, Óxido Nítrico, Álcool, Benzeno, Dióxido de Carbono e Fumo	https://www.ptrobotics.com/atmosfericos/6226-mq-135-air-quality-sensor-detection-module.html	6,03€	https://www.olimex.com/Products/Components/Sensors/Gas/SNS-MQ135/resources/SNS-MQ135.pdf
Qualidade do Ar	CCS811	Mede TVOC, eCO2 e MOX	https://www.ptrobotics.com/atmosfericos/5103-sparkfun-air-quality-breakout-ccs811.html	29,52€	https://cdn.sparkfun.com/assets/learn_tutorials/1/4/3/CCS811_Datasheet-DS000459.pdf
Qualidade do Ar	SGP30	Mede TVOC e eCO2	https://mauser.pt/catalog/product_info.php?cPath=1667_2669_2679&products_id=096-6749	20,20€	https://wiki.seeedstudio.com/Grove-VOC_and_eCO2_Gas_Sensor-SGP30/
Detetor de CO e Metano	MQ-9	Deteção de Monóxido de Carbono, Metano	https://www.ptrobotics.com/sensores-de-gases/6233-mq-9-co-carbon-monoxide-methane-liquefied-gas-sensor.html	7,75€	http://www.haoyuelectronics.com/Attachment/MQ-9/MQ9.pdf
Detetor de Hidrogenio	MQ-8	Deteção de Hidrogenio	https://www.ptrobotics.com/sensores-de-gases/6049-mq-8-hydrogen-gas-sensor-module.html	7,75€	https://dlnmh9ip6v2uc.cloudfront.net/datasheets/Sensors/Biometric/MQ-8.pdf
Detetor de CO	MQ-7	Deteção de Monóxido de Carbono	https://www.ptrobotics.com/sensores-de-gases/6232-mq-7-co-carbon-monoxide-coal-gas-sensor-module.html	7,13€	https://www.pololu.com/file/0J313/mq7.pdf%0A%0A
Detetor de LPG, Butano e Propano	MQ-6	Deteção de gás de petróleo liquido, butano e propano	https://www.ptrobotics.com/sensores-de-gases/6231-mq-6-liquefied-isobutane-propane-sensor-module.html	5,84€	https://www.pololu.com/file/0J312/MQ6.pdf

Detetor de Gás Natural e LPG	MQ-5	Deteção de gás natural e gás de petróleo liquido	https://www.ptrobotics.com/sensores-de-gases/6230-mq-5-liquefied-gas-methane-gas-sensor-module.html	4,24€	https://files.seeedstudio.com/wiki/Grove-Gas_Sensor-MQ5/res/MQ-5.pdf
Detetor de Gás Natural	MQ-4	Deteção de gás natural	https://www.ptrobotics.com/sensores-de-gases/6229-mq-4-natural-gas-methane-gas-sensor-module.html	6,95€	https://www.pololu.com/file/0J311/MQ4.pdf
Detetor de Alcool e Etanol	MQ-3	Detetor de álcool e etanol	https://www.ptrobotics.com/sensores-de-gases/6228-mq-3-alcohol-ethanol-detection-sensor-module.html	6,27€	https://www.pololu.com/file/0J310/MQ3.pdf
Detetor de Metano, Butano e LPG	MQ-2	Detetor de Metano, Butano e LPG	https://www.ptrobotics.com/sensores-de-gases/6227-mq-2-smoke-gas-lpg-butane-hydrogen-sensor-module.html	4,24€	https://www.pololu.com/file/0J309/MQ2.pdf
Microfone	Whadda WPSE309	Saida analógica ou saida digital para acusar quando som passa o limite definido.	https://mauser.pt/catalog/product_info.php?cPath=1667_2669_2685&products_id=096-4654	4,56€	https://www.velleman.eu/downloads/29/vma309_a4v02.pdf
Detetor de água		Deteta chuva ou nivel da água	https://www.ptrobotics.com/sensores-variados/4789-water-level-sensor.html	3,01€	
Detetor de água com aquecimento	Kemo M152	Quando deteta água/chuva liga uma resistência eletrica para secar mais fácilmente, ajudando com falsos alarmes causados por humidade	https://mauser.pt/catalog/product_info.php?cPath=1667_1544&products_id=096-5101	30,68€	https://www.kemo-electronic.de/en/House/Garden/M152-Rain-Sensor-12-V-DC.php
Encoder rotativo		Encoder rotativo com botão	https://mauser.pt/catalog/product_info.php?cPath=1667_2604_2682&products_id=096-7591	2,4€ (1 uni) / 2,34€ (5+ uni)	
Sensor UV	Adafruit VEML6070	Usa I2C e mede luz UV A	https://www.ptrobotics.com/sensores-opticos/4396-adafruit-veml6070-uv-index-sensor-breakout.html	9,35€	https://cdn-learn.adafruit.com/downloads/pdf/adafruit-veml6070-uv-light-sensor-breakout.pdf
Estação meteorologica		Mede velocidade do vento, direção do vento e quantidade de chuva	https://www.ptrobotics.com/atmosfericos/8596-estacao-meteorologica.html		
Barómetro/Temperatura	Adafruit BMP280	5V e 3.3V	https://www.ptrobotics.com/atmosfericos/9084-adafruit-bmp280-i2c-ou-spi-sensor-barometrico-de-pressao-e-altitude-stemma-qt.html	15,68€	https://cdn-shop.adafruit.com/datasheets/BST-BMP280-DS001-11.pdf
Barómetro/Temperatura	BMP280	3.3V	https://www.ptrobotics.com/sensor-de-pressao/8340-modulo-sensor-de-pressao-barometrica-gy-bmp280-33.html	2,71€	https://cdn-shop.adafruit.com/datasheets/BST-BMP280-DS001-11.pdf
Atuadores					
Tipo	Nome	Descrição	Link loja Portugal	Preço	Link especificações
Campainha/Buzzer	Buzzer Ativo	5V	https://www.ptrobotics.com/som/6108-active-buzzer-5v-12x95mm.html	0,98€	
Campainha/Buzzer	Buzzer Passivo	5V	https://www.ptrobotics.com/som/6109-passive-buzzer-5v-12x85mm.html	0,98€	
Servo	SG90	180º 1.6kg/cm(4,8V)	https://mauser.pt/catalog/product_info.php?cPath=324_2610_2612&products_id=096-6477	2,85€	
Relé		5V com furação	https://mauser.pt/catalog/product_info.php?cPath=324_517_2118&products_id=096-7589	2,65€ (1 uni) / 2,55€ (5+ uni)	
Relé		5V sem furação	https://mauser.pt/catalog/product_info.php?cPath=324_517_2118&products_id=096-7804	1,25€ (1 uni) / 0,80€ (5+ uni)	
Componentes Extra					
Tipo	Nome	Descrição	Link loja Portugal	Preço	Link especificações
Logic level converter	Conversor de nível lógico bi-direcional	Converter sensores de 5V para 3.3V	https://www.botnroll.com/pt/conversores-nivel-logico/3003-conversor-de-n-vel-l-gico-3-3v-5v-4-canais.html	1,50€	