Universidade de Aveiro
2021

BRUNO EDGAR ALMEIDA PRAZERES

BOMBA DE CALOR – MONITORIZAÇÃO EM TEMPO REAL

HEAT PUMPS – REAL TIME MONITORING

"Electric cars are really cool but air sources heat pumps are great."

David John Cameron

Universidade de Aveiro
2021

BRUNO EDGAR ALMEIDA PRAZERES

BOMBA DE CALOR – MONITORIZAÇÃO EM TEMPO REAL

HEAT PUMPS – REAL TIME MONITORING

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Mecânica, realizada sob a orientação científica do Doutor José Paulo Oliveira Santos, Professor Auxiliar do Departamento de Engenharia Mecânica da Universidade de Aveiro e coorientador Professor Vítor António Ferreira da Costa Professor Catedrático do Departamento de Engenharia Mecânica da Universidade de Aveiro.

Este trabalho teve o apoio financeiro dos projetos UIDB/00481/2020 e UIDP/00481/2020 - FCT - Fundação para Ciência e Tecnologia; e CENTRO-01-0145-FEDER-022083 - Programa Operacional Regional do Centro (Centro2020), no âmbito do Acordo de Parceria Portugal 2020, através do Fundo Europeu de Desenvolvimento Regional.

O Júri

Presidente	Prof. Doutor Nelson Amadeu Dias Martins Professor Associado da Universidade de Aveiro
Vogal / Arguente Principal	Prof. Doutor Rui Manuel Escadas Ramos Martins Professor Auxiliar da Universidade de Aveiro
Vogal / Orientador	Prof. Doutor José Paulo Oliveira Santos Professor Auxiliar da Universidade de Aveiro

Agradecimentos

Agradeço em primeiro lugar à minha família pelo apoio incondicional em toda a minha vida e pelos valores incutidos que me tornaram na pessoa que sou.

Um especial obrigado a todos os amigos que de uma forma ou de outra contribuíram para que tenha chegado até esta etapa, porque sem eles o caminho teria sido no mínimo mais complicado.

Agradecimentos ao meu orientador, Professor Doutor José Paulo Santos, e ao meu coorientador, Professor Catedrático Vítor António Ferreira da Costa pelos conselhos e diretrizes para a execução desta dissertação. Obrigado ao Engenheiro Virgílio Araújo pelo auxílio prestado no Laboratório de Termofluidos do Departamento de Engenharia Mecânica.

De um modo geral, muito obrigado a todos os que influenciaram o meu percurso académico pois sem eles não estaria aqui.

Palavras-chave

Monitorização, Bomba de Calor, Internet of Things (IoT), Comunicações Machine-to-Machine (M2M), Arduino, Raspberry pi, Algoritmo preditivo, Human-Machine Interface (HMI).

Resumo

A monitorização é uma área da engenharia que tem vindo a crescer nas últimas décadas, não só devido à complementaridade que acrescenta às soluções dos problemas que surgem nos projetos, como também é essencial para que a operação seja continuamente controlada e supervisionada.

A necessidade de monitorizar os equipamentos nunca foi tão presente, e para isso é preciso analisar cada um individualmente e escolher os parâmetros mais significativos.

Este trabalho incide-se na monitorização de uma bomba de calor por compressor de vapor. Trata-se de uma máquina termodinâmica que está em grande crescimento no mercado do aquecimento de águas sanitárias e de ambientes interiores. Depois de estudar o equipamento foram escolhidos quais os parâmetros a recolher do sistema e quais os dispositivos a utilizar para a sua monitorização.

No decorrer dos trabalhos, após a escolha dos parâmetros a analisar, reparou-se que havia a possibilidade de usar dois controladores distintos, e então, para além de apresentar a solução para o problema proposto, foram comparadas duas formas de resolução, apontando os vários aspetos positivos e negativos de cada uma delas.

Adicionalmente, foi implementado um algoritmo preditivo de falhas que consegue, antecipadamente, dar o alerta de que algo no equipamento está a tender para os limites não estipulados, podendo assim contribuir para a sua manutenção preditiva.

No final são apresentadas interfaces de controlo da bomba de calor para que o utilizador tenha o poder de ligar e desligar o aparelho, supervisionar os parâmetros recolhidos, e se algum problema surgir ser imediatamente notificado e informado das suas possíveis causas.

O documento assumiu uma forma didática, dando a conhecer as ferramentas que existem para que seja possível tornar qualquer projeto inteligente, seja ele comercial, doméstico ou industrial.

Keywords

Monitoring, Heat Pump, Internet of Things, Machine-to-Machine Communications (M2M), Arduino, Raspberry pi, Predictive Algorithm, Human-Machine Interface (HMI).

Abstract

Monitoring is an engineering area that has been growing in the last decades, not just because the benefices that add to a solution, but also because it's essential to have an operation continuously controlled and supervised.

The need to monitor equipment has never been more present, and for this it is necessary to analyse each one individually and choose the most significant parameters.

This work focuses on monitoring a steam compressor heat pump. It is a thermodynamic machine that is vastly growing in the indoors and domestic water heating market. After studying the equipment, it was time to choose the system parameters to be collected and the devices to be used for monitoring.

During work, after choosing the parameters to be analysed, it was noticed that there was the possibility of using two different controllers, and then, in addition to presenting the solution to the proposed problem, two forms of resolution were compared, pointing out the various positive and negative aspects of each one.

Additionally, a failure predictive algorithm was implemented, which can, in advance, alert that something in the equipment is tending towards the stipulated limits, thus contributing to its predictive maintenance.

At the end, the heat pump control interfaces are presented so that the user can turn the appliance on and off, supervise the collected parameters and if any problem arise, be immediately notified, and informed of its possible cause.

The document took a didactic form, introducing the tools that exist to make any project intelligent, whether commercial, domestic, or industrial.

Abreviaturas e siglas

ACK
ACKnowledge do Slave

ADC
Analog to Digital Converter

cmd
Interpretador de linhas de comando

CS
Chip Select

EEPROM
Electrically Erasable Programmable Read Only Memory

ESP
Microcontrolador ESP8266

F
Fim de mensagem I2C

GPIO
General-Purpose Input/Output

GPS
Global Positioning System

HMI
Human Machine Interface

HTTP
HyperText Transfer Protocol

I
Início de mensagem I2C

IoT
Internet of Things

IP
Internet Protocol

I2C
Inter Integrated Circuit

KΩ
Kiloohm

MQTT
Message Queuing Telemetry Transport

SQL
Structured Query Language

M2M
Machine to Machine

NACK
Acknowledge do Master

NTC
Negative Temperature Coefficient

OS
Operative System

PHP
Hypertext Preprocessor

RAM
Random Access Memory

ROM
Read Only Memory

SCL
Serial Clock

SDA
Serial Data

SMBus
System Management Bus

SPI
Serial Periferal Interface

TCP
Transmission Control Protocol

UDP
User Datagram Protocol

UML
Unified Modeling Language

URL
Uniform Resource Locator

USB
Universal Serial Bus

VNC
Virtual Network Computing

WiFi
Wireless Fidelity

Índice

Abreviaturas e siglas xiii
Índice xv
Lista de Figuras xvii
Lista de Tabelas xix
1. Introdução 1
1.1. Enquadramento 2
1.2. Objetivos do trabalho 3
2. Estado da Arte 5
2.1. Monitorização 6
2.2. *Internet of Things* 6
2.3. Controladores Arduino e Raspberry pi 7
2.4. Protocolos de comunicação 7
2.4.1. I2C 8
2.4.2. SPI 9
2.4.3. TCP/IP 10
2.4.4. MQTT 12
2.4.5. HTTP 12
2.5. Algoritmos preditivos 13
2.6. Ponto de partida dos trabalhos 13
3. Bomba de Calor por Compressão de Vapor 15
3.1. Ciclo termodinâmico 16
3.2. Bomba de calor em estudo 17
3.3. Problemas mais comuns de uma bombas de calor e respetivas causas 20
4. *Hardware* e *Software* implementado 21
4.1. *Hardware* 23
4.1.1. Sensores 23
4.1.3. Controlador do sistema 30
4.2. Software 35
4.2.1. Arduino no microcontrolador ESP8266 35
4.2.2. *Python* no Raspberry pi 4 40
4.2.3. Human Machine Interface 43
4.2.4. Algoritmo preditivo 47
5. Resultados 51
5.1. Fluxogramas das soluções propostas 52
5.1.1. Solução para o microcontrolador ESP8266 52
5.1.2. Solução para o microcomputador Raspberry pi 4 54
5.2. Previsões do algoritmo 56
5.3. Interface *Web* 57
5.3.1. *Login* 57

5.3.2. Monitorização 58
5.3.3. Supervisão 59
5.3.4. Histórico 60
5.3.5. Falhas e prováveis causas 61
5.3.6. Notificações 63
5.4. Interface *Android* 64
5.5. Comparação final 65
6. Conclusões 67
Bibliografia 71

Lista de Figuras

Figura 1: Exemplo das variações de tensões nos condutores com o protocolo SPI, [5]. 9
Figura 2: Esquema de mensagem TCP/IP, [6]..... 10
Figura 3: Protocolos utilizados em cada camada da mensagem TCP/IP, [6]..... 11
Figura 4: Exemplo de endereço URL. 12
Figura 5: -Esq.: Representação esquemática de uma bomba de calor por compressão de vapor; -Dir.: Representação do ciclo termodinâmico no diagrama T-s; [9]..... 16
Figura 6: Bomba de calor a monitorizar: (1) Render do protótipo da bomba de calor; (2) Desenho 3D do interior da bomba de calor; (3) Protótipo físico da bomba de calor, [10]. 17
Figura 7: Representação detalhada da bomba de calor: (1) Compressor; (2) Sensor de pressão, filtro secador e válvula de expansão;(3) Ventilador e evaporador; (4) Condensador / Permutador de calor para o meio a aquecer, [10]. 17
Figura 8: Esquema da disposição dos componentes da bomba de calor..... 19
Figura 9: Esquema da disposição dos componentes da bomba de calor e dos componentes do sistema desenvolvido..... 22
Figura 10: Sensor GY-521 MPU6050 24
Figura 11: Sensor TC-74 25
Figura 12: Sensor MAX6675 K..... 27
Figura 13: Esquema elétrico do Módulo Relé 28
Figura 14: Modulo Relé..... 29
Figura 15: Microcontrolador ESP8266MOD. 30
Figura 16: Microcomputador Raspberry pi 4, Model B. 32
Figura 17: Esquema elétrico dos sensores e modulo relé utilizados com o microcontrolador ESP8266..... 33
Figura 18: Esquema elétrico dos sensores e modulo relé utilizados com o microcomputador Raspberry pi 4.34
Figura 19: Ambiente de desenvolvimento Arduino..... 35
Figura 20: Processamento da mensagem SPI proveniente do MAX6675 41
Figura 21: Evolução temporal da temperatura do compressor. 48
Figura 22: Evolução temporal da temperatura do compressor com a reta preditiva..... 49
Figura 23: Fluxograma da solução para o microcontrolador ESP8266. 53
Figura 24: Fluxograma da solução para o microcomputador Raspberry pi 4. 54
Figura 25: Página de Login da interface Web. 57
Figura 26: Página de Monitorização da interface Web..... 58
Figura 27: Página de Supervisão da interface Web. 59
Figura 28: Página de Histórico da interface Web. 60
Figura 29: Página de Falhas e prováveis causas da interface Web. 61
Figura 30: Interface Android 64

Lista de Tabelas

Tabela 1: Cronologia esquemática das mensagens do protocolo I2C, [4]. 8
Tabela 2: Posições de memória e respetivos endereços do sensor GY-521 MPU6050. 25
Tabela 3: Posições de memória do sensor TC74 e respetivos endereços, [14]. 26
Tabela 4: Funções da biblioteca Wire e respetiva descrição. 36
Tabela 5: Funções da biblioteca max6675 e respetiva descrição. 37
Tabela 6: Funções da biblioteca esp8266wifi e respetiva descrição. 38
Tabela 7: Funções da biblioteca PubSubclient e respetiva descrição. 39
Tabela 8: Funções da biblioteca SMBus e respetiva descrição. 40
Tabela 9: Funções da biblioteca spidev e respetiva descrição. 41
Tabela 10: Funções da biblioteca “paho.mqtt.publish” e respetiva descrição 42
Tabela 11: Nodes utilizados para processar os dados no Node-Red e respetiva descrição. 44
Tabela 12: Nodes utilizados na construção do Dashboard e respetiva descrição. 45
Tabela 13: Intervalos limite de temperaturas usados no algoritmo preditivo. 56
Tabela 14: Mensagens do histórico, falhas e prováveis causas. 62
Tabela 15: Notificações pop-up da interface Web. 63
Tabela 16: Análise comparativa das duas soluções propostas. 66

Capítulo I

Introdução

1.1. Enquadramento

O aquecimento de águas sanitárias é imprescindível para um maior conforto e qualidade de vida. Com o desenvolvimento tecnológico, hoje em dia, é possível encontrar soluções no mercado que proporcionam água quente, como caldeiras, esquentadores e termoacumuladores. A maioria utiliza combustíveis fósseis, e consumos energéticos elevados, contribuindo, assim, para que o aquecimento de águas sanitárias seja um grande consumidor energético, uma vez que um simples banho pode consumir 5kWh de energia, [1]. Este problema incentiva a exploração e otimização de soluções alternativas, dando força ao mercado das bombas de calor.

As bombas de calor são equipamentos de aquecimento bastante atrativos, uma vez que permitem obter um efeito útil de aquecimento várias vezes superior ao *input* energético necessário para o seu funcionamento. Adicionalmente, como se evolui para um cenário de energia elétrica abundante de origem renovável, este é outro dos argumentos a favor da utilização das bombas de calor (que geralmente são alimentadas eletricamente).

A monitorização em tempo real do funcionamento destes equipamentos (por recolha de informação a partir dos seus constituintes mais relevantes) assume uma importância cada vez maior, para, por exemplo: avaliar o bom ou deficiente funcionamento do equipamento, dar indicações sobre o funcionamento deficiente de algum dos seus componentes, permitir a integração dialogante da bomba de calor em plataformas de climatização mais abrangentes, permitir a integração das bombas de calor em sistemas *Smart*, permitindo comunicação entre dispositivos, definir padrões de utilização e adaptando-se, tendo a possibilidade de controlo a partir de um só aparelho, entre outros. Para tal, a recolha dos parâmetros de funcionamento da bomba de calor em tempo real, e a sua comparação com os padrões expectáveis quando em funcionamento normal, permite a geração de um vasto conjunto de informação que pode ser armazenada e tratada e enviada para determinados destinatários com várias finalidades, como emitir alarmes ou mensagens de mau funcionamento, com a possibilidade de indicar as causas mais prováveis para esse mau funcionamento.

No mercado podem ser encontradas várias marcas que ao venderem a bomba de calor disponibilizam uma aplicação *mobile*, com o objetivo de controlar modos de operação e temperaturas desejadas. Estas interfaces são dedicadas ao utilizador, e como tal, são muito limitadas na deteção de falhas, havendo a necessidade de as completar com informações mais técnicas, não tanto para o comprador, mas para os técnicos de manutenção, facilitando a abordagem às possíveis avarias.

1.2. Objetivos do trabalho

O presente trabalho tem como principal objetivo o desenvolvimento e implementação do sistema de rastreabilidade de dados de uma bomba de calor adicionando diversos sensores de temperatura e vibração à máquina. Em seguida, disponibilizar esses dados de forma a prever possíveis avarias, tanto na bomba de calor como nos sensores, e, consequentemente, agir da melhor forma para evitar danos permanentes no equipamento e na segurança dos utilizadores. Estas ações passam pelo aviso da bomba de calor ao utilizador e técnico instalador do mau funcionamento, requisitando a sua intervenção e podendo interromper o funcionamento do sistema de forma imediata.

Além de obter as variáveis em tempo real, estas são armazenadas numa base de dados de modo a gerar um histórico para fornecer à empresa de manutenção e, por fim, diagnostica o problema da melhor forma. Toda a informação recolhida tem também o objetivo de ser adaptada aos dispositivos móveis através de uma aplicação interativa, permitindo a capacidade de monitorização em qualquer lugar do planeta com ligação à Internet.

Capítulo II

Estado da Arte

2.1. Monitorização

A monitorização é uma forma de acompanhar um certo sistema, fornecendo ao utilizador informações contínuas do estado do equipamento, por aplicação de sensores e emissão de alertas de combinações de eventos. O primeiro projeto prático de monitorização teve lugar em meados do século IXX, mantendo uma comunicação entre o palácio do Imperador Russo e a sede do exército. Posteriormente, através de termopares e sensores profundidade instalados no *Mont Blanc* foi possível saber em Paris em tempo real a temperatura e a altura da neve. Apesar destas soluções fornecerem informações importantes com alguma exatidão havia sempre bastante ruído nos dados que, muitas vezes, podiam provocar leituras incorretas, [2].

Ao longo dos anos, a monitorização foi crescendo e ganhando força em vários ramos, tanto na indústria como nas habitações, tornando as ligações por cabo mais fiáveis e seguras e muitas delas foram evoluídas para ligações *wireless*. Esta evolução permitiu desenvolver ferramentas para transmitir dados entre humanos e máquinas, através de interfaces HMI – Human Machine Interface e, futuramente, entre máquinas, M2M – Machine to Machine, de uma maneira fácil e eficaz. Este avanço tecnológico deu origem ao conceito, bastante comum nos dias de hoje, de *Internet of Things*.

2.2. *Internet of Things*

A IoT – Internet of Things, como o nome indica, é a capacidade dos objetos eletrónicos que nos rodeiam se ligarem á Internet e, assim estabelecerem contacto entre si. Esta conexão permite uma comunicação permanente entre o estado dos equipamentos e o utilizador, sendo que, no início desta década, é praticamente impossível encontrar uma pessoa que não esteja em contacto com dezenas de equipamentos eletrónicos, tais como simples lâmpadas, automóveis e sistemas de ar condicionado. Juntando ao intenso mercado de *smartphones* na sociedade, é de notar que cada pessoa tem a sua própria interface humana-máquina com acesso à Internet constantemente a menos de um metro de distância. Isto abre imensas portas para o mundo da monitorização em tempo real, pois basta implementar dispositivos relativamente económicos a equipamentos eletrónicos para ter o poder de controlar aparelhos em qualquer parte do mundo. A internet das coisas abre, assim, os horizontes em relação à comodidade e conforto de monitorização e controlo de equipamentos e sistemas através de simples cliques.

2.3. Controladores Arduino e Raspberry pi

No mundo da eletrónica e da informática somos constantemente confrontados com as rivalidades entre marcas: nos sistemas operativos o Windows contra a Apple, nos processadores a Intel contra a AMD, e agora o Arduino contra o Raspberry pi. Estes últimos dois dispositivos são constantemente comparados dado possuírem tamanhos de hardware similares, serem fáceis de ligar componentes elétricos aos seus pinos GPIO, como motores, leds, sensores, entre outros, e estarem a ganhar terreno no mundo dos controladores.

Mas esta comparação não poderia ser mais inadequada para o papel que cada um consegue desempenhar, uma vez que estamos a comparar um microcontrolador com um microcomputador. O Arduino ESP8266 é ideal para controlar pequenos componentes eletrónicos e muito útil para gerir as operações de um robô, tendo recursos como *WiFi* e a possibilidade de adicionar *Bluetooth*, *Ethernet* e outros meios de comunicação. Conta com um microcontrolador onde guarda e corre os programas, deixando de ser necessárias memórias externas e outros componentes, assim contribuindo para um baixo consumo energético. Por via de um ambiente de trabalho intuitivo, utilizando linguagem C e as inúmeras bibliotecas já desenvolvidas, o Arduino é bastante simples de programar, apesar de depender de um computador para modificar e criar o código, [3]. Por outro lado, o Raspberry Pi é considerado ser um microcomputador, pois possui um microprocessador que, para funcionar, requer outros componentes como memórias RAM e ROM, placa gráfica e outros. Ou seja, é um computador com sistema operativo Linux, onde o programa pode ser escrito diretamente no dispositivo, podendo também usar ferramentas de programação, armazenamento e tratamento de dados mais avançadas, contribuindo para uma operação mais complexa, [4]. O Raspberry Pi consegue criar servidores *Web*, armazenar dados, utilizar câmaras Web e estabelecer ligações USB, ao contrário do Arduino, que se limita a controlar componentes eletrónicos e a correr os programas previamente carregados.

2.4. Protocolos de comunicação

O termo protocolo alberga inúmeros significados no mundo moderno. Na área da eletrónica e de computadores prende-se a um conjunto de regras que permitem criar um formato de mensagem e enviá-la para outro equipamento, sem que haja interferências ou falhas na comunicação. Existe mais do que um protocolo para cada uma das variadíssimas aplicações, e por isso, neste subcapítulo serão apenas abordados os protocolos de comunicação utilizados no protótipo final desenvolvido.

2.4.1. I2C

O protocolo *Inter Integrated Circuit* é um modo de comunicação adotado por circuitos como EEPROM's, sensores de temperatura, microcontroladores, GPS, entre outros. Consiste na comunicação entre *Master* e *Slave*, mas agora com apenas dois condutores dedicados à comunicação. O primeiro é utilizado para transmitir o sinal de relógio, *Serial Clock* - SCL. Isto serve para definir uma frequência entre dispositivos, para que os dois estejam sincronizados no envio e receção de mensagens, adquirindo o nome de comunicação síncrona. O responsável por emitir este sinal é o *Master*. O segundo condutor é por onde a mensagem será enviada e recebida, ou seja, é bidirecional e tem o nome de *Serial Data* - SDA. O *Master* é quem controla a linha de dados a maior parte do tempo, enviando ordens e dados para os *Slaves*, tendo cada um o seu próprio endereço de sete bits. Estes podem apenas guardar essas informações ou responder, e é neste momento que lhes é concedido, pontualmente, o controlo do SDA para transmitir dados pedidos.

A mensagem I2C tem caraterísticas particulares para que haja entendimento entre dispositivos. Existem duas estruturas principais para a mensagem, uma para a escrita e outra para a leitura, que são apresentadas na Tabela 1.

Esquema da Mensagem de Escrita								
I	Endereço do *Slave*	Bit de escrita (0)	ACK	Endereço da memória	ACK	Dados	ACK	F
	7 bits	1 bit		8 ou 16 bits		8 bits		

Esquema de Mensagem de Leitura												
I	Endereço do *Slave*	Bit de escrita (0)	ACK	Endereço da memória	ACK	I	Endereço do *Slave*	Bit de Leitura (1)	ACK	Dados	NACK	F
	7 bits	1 bit		8 ou 16 bits			7 bits	1 bit		8 bits		

Tabela 1: Cronologia esquemática das mensagens do protocolo I2C, [5].

Em primeiro lugar, é importante referir que quando a comunicação está inativa os dois condutores têm tensões iguais à tensão gerada no *Master*, Vdd. Para se dar o início à mensagem de leitura é preciso forçar a tensão do SDA a 0 V e posteriormente o SCL, evento I. A seguir é essencial indicar qual o *Slave* que irá comunicar, sendo enviado o endereço binário correspondente, indicando que irá escrever ou ler. Este responde forçando a linha de dados a 0 V, evento ACK, fazendo-o sempre que recebe oito bits, e espera que o *Master* selecione a posição de memória que quer ler ou alterar. Depois, se a mensagem for de escrita, são enviados os dados e termina a comunicação,

libertando a tensão no sinal de relógio e a seguir no sinal de dados de forma a ficarem com tensão igual a Vdd, evento F. Se a mensagem for de leitura, após selecionar o endereço de memória tem de se fazer um novo início de mensagem, evento I, e enviar novamente o endereço do *Slave*, mas agora com um bit de leitura. De seguida são enviados para o *Master* os bits que se encontravam na posição de memória previamente selecionada, e é finalizada a comunicação. O evento NACK indica que a mensagem enviada pelo *Slave* terminou e é enviado pelo *Master*.

Este exemplo serve apenas para explicar a comunicação entre dispositivos com vários endereços de memória que utilizam este protocolo. Existem outros equipamentos sem este nível de complexidade, que apenas tem um endereço, fazendo com que as mensagens se tornem mais simples.

2.4.2. SPI

O protocolo *Serial Peripheral Interface,* à semelhança do anteriormente abordado, serve para conectar fisicamente dois dispositivos para que possam comunicar entre si. Este protocolo é usado por sensores de temperatura e pressão, ADC, EEPROM's, entre muitos outros dispositivos. O modo de funcionamento baseia-se na atribuição de um *Master*, normalmente o controlador, e um ou mais *Slaves*. Para a comunicação é necessário utilizar dois *Serial Data*, um para receber, SDA_{in}, e outro, SDA_{out}, para enviar os dados, um *Serial Clock*, para definir a velocidade de envio e garantir a sincronização, e ainda um *Chip Select* - CS que irá indicar qual o equipamento que deverá responder à mensagem, deixando de lado os endereços do protocolo I2C. Uma particularidade deste protocolo é que o SDA não é bidirecional, e é, por isso, necessário um condutor para enviar e outro para receber. Sempre que o *Master* envia algo, o *Slave* responde mesmo que não tenha que enviar nada, completando assim um diálogo *full-duplex*.

A topologia de comunicação começa com a ativação do CS para selecionar o *Slave* que terá de responder, uma vez que é possível ter vários *Slave* num barramento. Depois, o SCL começa a alternar a tensão entre 0 V e Vdd, e o SDA envia a palavra de oito bits. Na imagem da figura 1 é possível observar graficamente estas variações de tensão nos condutores, [6].

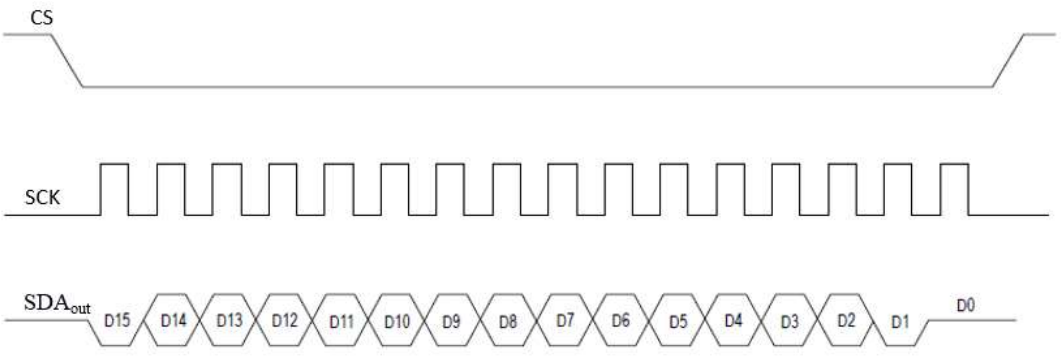


Figura 1: Exemplo das variações de tensões nos condutores com o protocolo SPI, [6].

Neste exemplo, o CS é forçado a 0V para que o dispositivo seja acionado, e inicia-se a variação periódica do SCK para que haja sincronização dos equipamentos. De seguida, procede-se ao envio dos dados a partir do SDA_{out}, onde podem ser transmitidos 16 bits de informação.

2.4.3. TCP/IP

No TCP/IP existem três protocolos principais, o *Transmission Control Protocol*, o *Internet Protocol* e o *User Datagram Protocol,* que cooperam entre si para que, mais uma vez, se estabeleça um meio de comunicação entre dispositivos. O TCP é utilizado para dispositivos que precisam de estabelecer uma conexão antes de enviar informação. É constituído por uma série de regras que estabelecem uma ligação lógica entre dispositivos antes de ser gerada qualquer mensagem. Já o UDP e IP não necessitam dessa prévia conexão, o que lhes permite obter uma melhor performance dinâmica.

Após estabelecer a conexão é gerada uma mensagem com formato único e organizada em três partes, *Header* – que contém informação sobre os dados da mensagem, ou como devem ser interpretados ou usados, *Data* – onde se encontra a informação que se quer transmitir, e que muitas vezes adquire o nome de *payload*, e *Footer* – similar ao início da mensagem e com parâmetros de controlo da mesma, sendo muitas vezes opcional. Como são usados vários protocolos a mensagem tem de ser organizada por quatro camadas, havendo vários *Headers* e *Footers* como ilustrado na imagem da figura 2.

Figura 2: Esquema de mensagem TCP/IP, [7].

As camadas que constituem este protocolo comunicação são *Network Interface*, *Internet*, *Transport* e *Application.* A *Network Interface,* como o nome indica, é a camada mais superficial e é onde o protocolo TCP/IP corre. A *Internet* é responsável pelo endereço dos dispositivos, armazenamento, manipulação e entrega dos dados e escolha do caminho da conexão. O *Transport* ou *Host-to-host* é onde encontramos as permissões para facilitar as conexões entre dispositivos. Por

fim, a *Application* é a camada mais profunda que é onde estão as funções criadas pelo utilizador. Na Figura 3 é possível observar os vários protocolos inseridos nas várias camadas. [7]

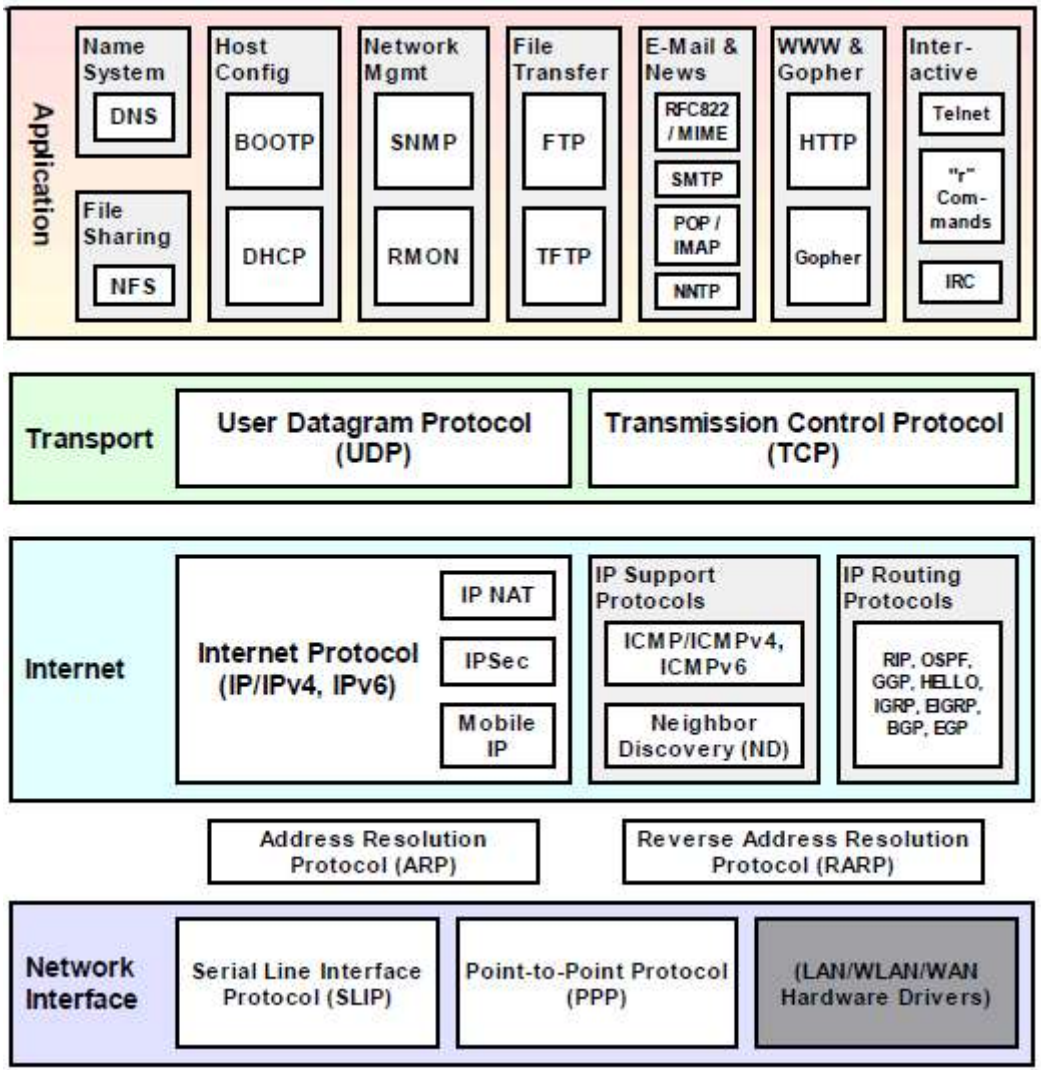


Figura 3: Protocolos utilizados em cada camada da mensagem TCP/IP, [7].

Pode-se verificar que cada mensagem possui diversos protocolos de comunicação, sendo importante referir que o protocolo IP se encontra na camada *Internet* e os protocolos UDP e TCP na camada *Transport*, sendo dos mais importantes para fazer chegar o corpo da mensagem presente na *Application* ao seu destinatário.

2.4.4. MQTT

O protocolo Message Queuing Telemetry Transport é o meio de comunicação criado para satisfazer necessidades impostas pela indústria da IoT. O protocolo foi projetado para ser um transporte de informação onde se publicam e subscrevem dados, ideal para conectar dispositivos com baixo poder de processamento e com reduzida largura de banda de rede. Outra caraterística é a sua capacidade de se conectar a milhares de dispositivos ao mesmo tempo, garantindo fiabilidade e segurança na troca de informação.

Este protocolo funciona a partir da utilização de um *broker* ou servidor, que vai armazenando as mensagens constituídas pelo *topic* e pela informação pretendida num documento do género base de dados presente no MQTT. O *topic* funciona como o nome de uma variável que vai guarda os dados. Isto é possível através do protocolo TCP/IP, que estabelece uma interação cliente-servidor com o *broker*.

2.4.5. HTTP

O protocolo *HyperText Transfer Protocol* serve para transferir documentos e ficheiros entre servidores e clientes *Web* recorrendo a um sistema de endereços denominado por *Uniform Resource Locator* - URL. É a pedra angular das comunicações *Web*, pois é a partir deste protocolo que são transferidas páginas em diversas linguagens como HTML, PHP ou *JavaScript*, sendo possível criar um número infinito de aplicações. Para haver pedidos HTTP é preciso estabelecer uma ligação TCP através do URL com o servidor *Web*. Este envia o documento pedido pelo mesmo caminho para que seja descodificado no *browser* local. Na Figura 4 é ilustrado um URL exemplo.

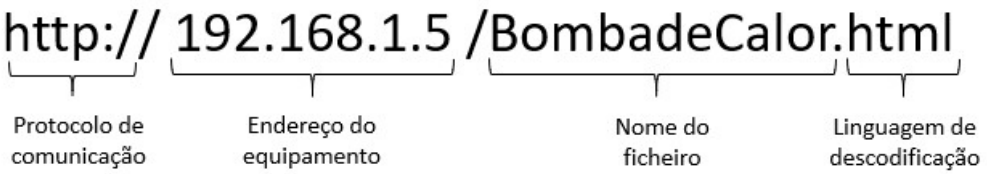


Figura 4: Exemplo de endereço URL.

As mensagens HTTP provenientes do cliente podem assumir três formas distintas: GET – serve para pedir ao servidor um documento, HEAD – apenas pede as propriedades de um ficheiro, e POST – pede novos dados para atualizar um documento. O servidor, para além de responder com o pedido, envia também códigos de estado da ligação ou do ficheiro. Por exemplo, a mensagem "HTTP/1.1 200" indica que está tudo operacional; já a mensagem "HTTP/1.1 404" significa que o ficheiro não foi encontrado no equipamento.

2.5. Algoritmos preditivos

Os algoritmos preditivos são um conjunto de operações que fazem uma análise preditiva da informação. Este conceito é sustentado pela criação e uso de modelos analíticos que fazem previsões de valores com base em padrões extraídos do histórico de dados. As aplicações para estes algoritmos são vastas, sendo usados por exemplo para prever preços de reservas em hotéis e de bilhetes de avião (para maximizar lucros), e para prever dosagens de medicamentos (para otimizar os tratamentos ou mesmo para auxiliar diagnósticos de doenças e falhas de equipamentos).

Estas aplicações têm sempre duas coisas em comum. A primeira é que todas usam modelos para guiar as suas previsões, e a segunda é que usam informação de inventos que já aconteceram para treinar esses modelos, sendo este treino denominado por *Machine Learning*, [8].

A "aprendizagem da máquina", mais conhecida por *Machine Learning* é uma área de estatística e inteligência artificial que se baseia em extrair conhecimentos de base de dados e estimar valores futuros. Existem três técnicas para tratar os dados e encontrar os modelos preditivos mais adequados para os vários problemas, como detetar falhas ou tomar decisões em tempo real ou parcelar clientes. Para o primeiro problema a aprendizagem supervisionada será a mais indicada, pois esta assemelha-se a uma regressão linear. Ou seja, é criada uma função com os valores previamente fornecidos para prever os valores seguintes. Se estes valores se afastarem dos valores padrão alguma falha surgirá brevemente.

Para a tomada de decisões a aprendizagem não supervisionada terá um papel fundamental, uma vez que permite catalogar os dados em classes. Isto é, os dados são relacionados e separados, podendo atribuir uma decisão a cada grupo de dados cumprindo os objetivos.

Por fim, existe a aprendizagem reforçada que se baseia num sistema de pontuação, sendo atribuída ou penalizada cada ação. Este cálculo contribui para definir qual o grupo mais adequado de um certo cliente, baseando-se nas suas escolhas, [9].

2.6. Ponto de partida dos trabalhos

Analisando o mercado atual da eletrónica e as pesquisas realizadas na área da monitorização é de notar que os avanços tecnológicos são constantes e de grande amplitude. Cada trabalho realizado neste tema prende-se a um objetivo específico como armazenar dados registados a partir de sensores, [2], ou criar algoritmos preditivos, [9], sendo invulgarmente aplicados em conjunto e a equipamentos comerciais. Por outro lado, as marcas presentes no mercado ainda não foram capazes de juntar estas investigações às suas soluções, criando apenas interfaces apelativas para o utilizador, mas pobres em aplicações, como deteção das causas dos erros ou mesmo a sua prévia deteção. Este trabalho consistiu em analisar as melhores soluções existentes e culminá-las todas num só sistema.

Capítulo III

Bomba de Calor por Compressão de Vapor

Sendo o foco deste trabalho a monitorização de uma bomba de calor, é necessário, em primeiro lugar, perceber com algum detalhe o que é este equipamento, para que serve, quais os estados normais de operação, quais os componentes essenciais e quais são os seus propósitos, para assim melhor entender quais os sensores e as suas posições no sistema que melhor se adequam à sua monitorização.

3.1. Ciclo termodinâmico

Uma bomba de calor por compressão de vapor opera segundo um ciclo termodinâmico, removendo o calor de uma zona de baixa temperatura e libertando o calor numa zona a mais alta temperatura. Para a sua operação, a bomba de calor recebe energia externa, sob a forma de trabalho ou de calor do ambiente.

Na Figura 5 é possível observar o ciclo termodinâmico da bomba de calor. O fluido operante inicia a sua aquisição de energia em forma de calor no evaporador, calor esse proveniente do ar ambiente, e é enviado para o compressor. Aí a pressão do fluido é elevada, e por sua vez a temperatura, também sendo transferido como vapor sobreaquecido para o condensador a alta pressão. Neste, é efetuada a permuta de calor para o meio a aquecer. Após a libertação da energia o fluido segue, como líquido saturado, para a válvula de expansão, onde diminui a sua pressão e temperatura, encontrando-se à saída da válvula de expansão a uma temperatura inferior á temperatura do ambiente. O fluido operante volta finalmente ao evaporador para completar o ciclo. Os diagramas de temperatura-entropia, (T-s), são muito úteis, pois com o auxílio de tabelas termodinâmicas é possível calcular pressões, temperaturas, volumes específicos, entalpias específicas e entropias específicas do fluido operante nos vários pontos do ciclo, [10].

O trabalho que é fornecido ao compressor geralmente a partir de um motor elétrico, força o escoamento do fluido operante da bomba de calor.

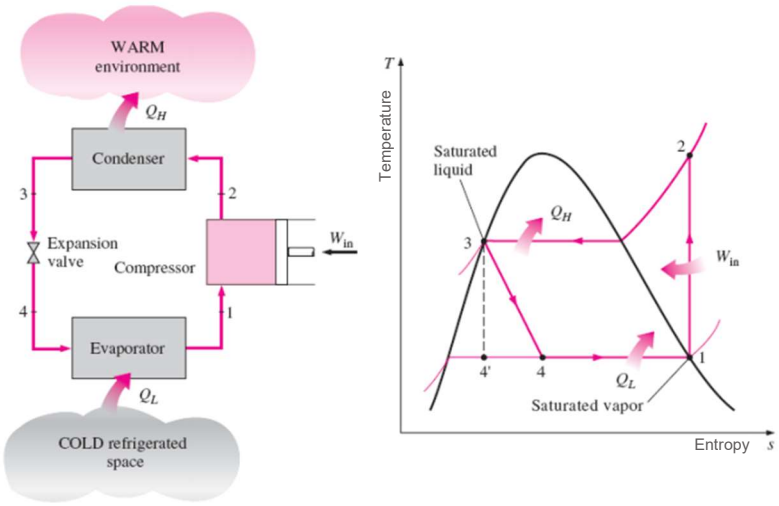


Figura 5: -Esq.: Representação esquemática de uma bomba de calor por compressão de vapor; -Dir.: Representação do ciclo termodinâmico no diagrama T-s; [10].

3.2. Bomba de calor em estudo

A bomba de calor utilizada para este trabalho foi um protótipo que a Bosch disponibilizou para o Laboratório de Termofluidos do Departamento de Engenharia Mecânica, com a designação comercial "Compress 3000 DWF Bosch" (que conta com algumas variantes do modelo, mas sempre com os mesmos princípios).

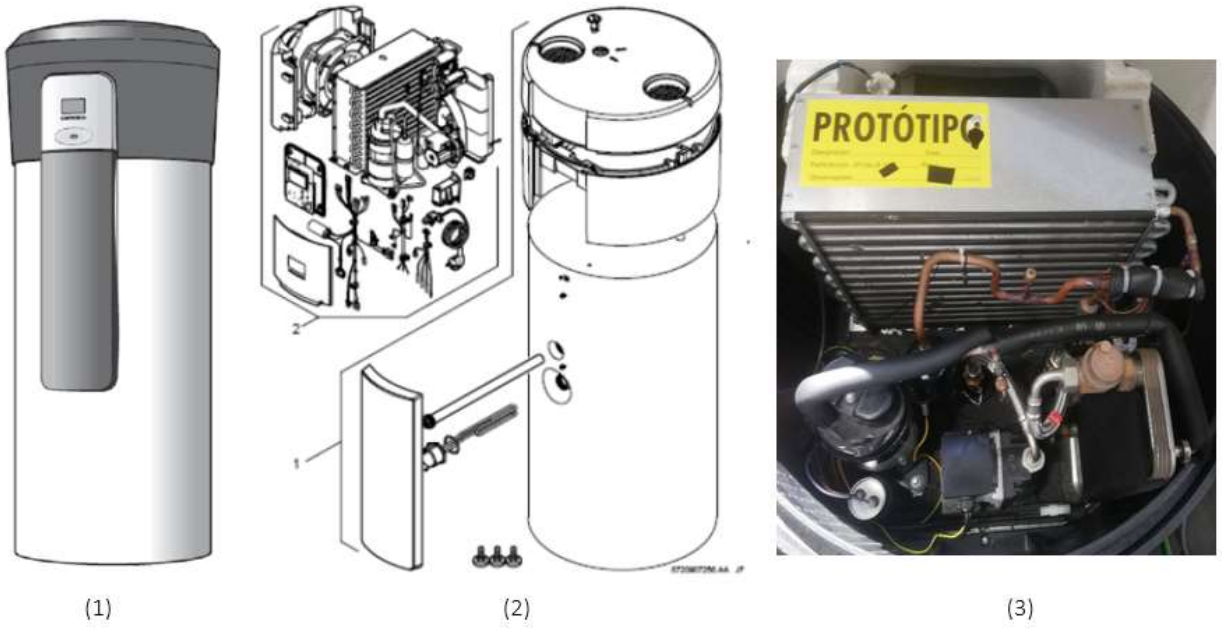


Figura 6: Bomba de calor a monitorizar: (1) Render do protótipo da bomba de calor; (2) Desenho 3D do interior da bomba de calor; (3) Protótipo físico da bomba de calor, [11].

Esta bomba de calor conta com vários componentes principais para o seu funcionamento, de onde se destacam o compressor, filtros secadores, a válvula de expansão, sensores de pressão, o evaporador com ventilador, o condensador/permutador e sensores de temperatura, [11].

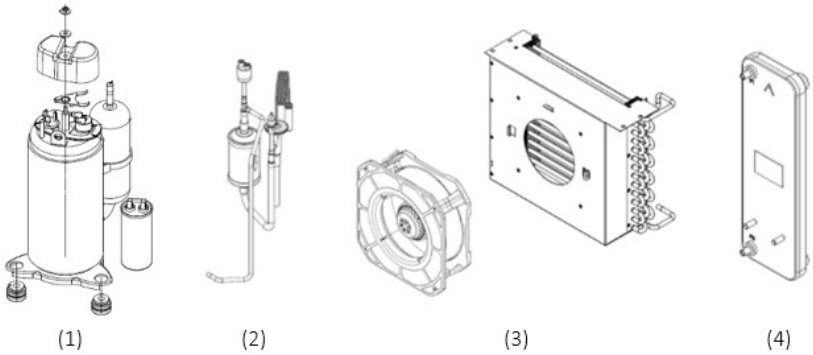


Figura 7: Representação detalhada da bomba de calor: (1) Compressor; (2) Sensor de pressão, filtro secador e válvula de expansão;(3) Ventilador e evaporador; (4) Condensador / Permutador de calor para o meio a aquecer, [11].

•Compressor

O compressor, acionado por um motor elétrico, comprime e força o escoamento do refrigerante através dos vários componentes do sistema. O seu principal papel é comprimir o refrigerante proveniente do evaporador que se encontra na forma de vapor a uma temperatura baixa, sendo a compressão acompanhada por um aumento de temperatura, a qual é maior que a temperatura do meio a aquecer. Em seguida é encaminhado, a alta temperatura e alta pressão, para o condensador. Um ponto importante é a necessidade de lubrificação do compressor por um óleo especial, que é essencial para a sua manutenção. Este protótipo foi instalado com um compressor da Mitsubishi Electric, o modelo KB134VFN, que necessita de 450 W de potência para o seu funcionamento, [12].

•Condensador/Permutador de calor

O condensador é onde ocorre a libertação da energia na forma de calor para o meio a aquecer. O refrigerante quente entra no condensador, liberta energia para o sistema de aquecimento do reservatório de água quente da bomba de calor e, à medida que cede calor, evolui de vapor sobreaquecido para líquido saturado.

•Filtros secadores

O vapor de água pode causar problemas fatais para os sistemas envolvendo fluidos refrigerantes, podendo causar o congelamento do orifício da válvula de expansão, corroendo e molhando o compressor e provocando a deterioração do motor e a formação de borras de óleo. O filtro secador protege a válvula de expansão, absorvendo a humidade que, por alguma razão, pode estar misturada com o fluido refrigerante.

•Válvula de expansão

A válvula de expansão tem dois objetivos, que são controlar a quantidade de refrigerante que circula na instalação e manter a diferença de pressão estável entre o condensador e o evaporador.

•Sensores/Controladores de pressão

Estes sensores permitem o controlo da pressão antes e depois do refrigerante passar pelo compressor, protegendo-o de temperaturas muito altas ou muito baixas. Se a pressão for muito baixa o compressor para, e apenas se permite o seu arranque após o *reset* do sistema e a leitura de pressão normal.

•Evaporador

O evaporador é o responsável pela transferência de calor entre o ar ambiente e o fluido refrigerante. Por outras palavras, evolui de uma mistura bifásica de líquido mais vapor saturado para o estado de vapor à medida que recebe calor.

•Sensores de temperatura

Os sensores utilizados são do tipo NTC, que são resistências que variam inversamente com a temperatura a que são sujeitas. Neste protótipo estão presentes quatro sensores deste tipo, dos quais três têm uma resistência de 10 KΩ e um de 8,5 KΩ quando estão a 25°C. Os NTC de maior resistência foram colocados na entrada de ar exterior no evaporador e na entrada e saída de água do condensador, e o de menor resistência na saída de ar do evaporador. Os valores medidos por estes sensores podem variar entre -10°C a 100°C e é através deles que se realiza o controlo da bomba de calor, uma vez que estes são enviados para uma interface física HMI e o utilizador pode variar a temperatura desejada do reservatório de água quente, e ser ativado automaticamente o modo de anti congelamento, entre outras ações. Na Figura 8 é apresentado um esquema com a disposição dos componentes.

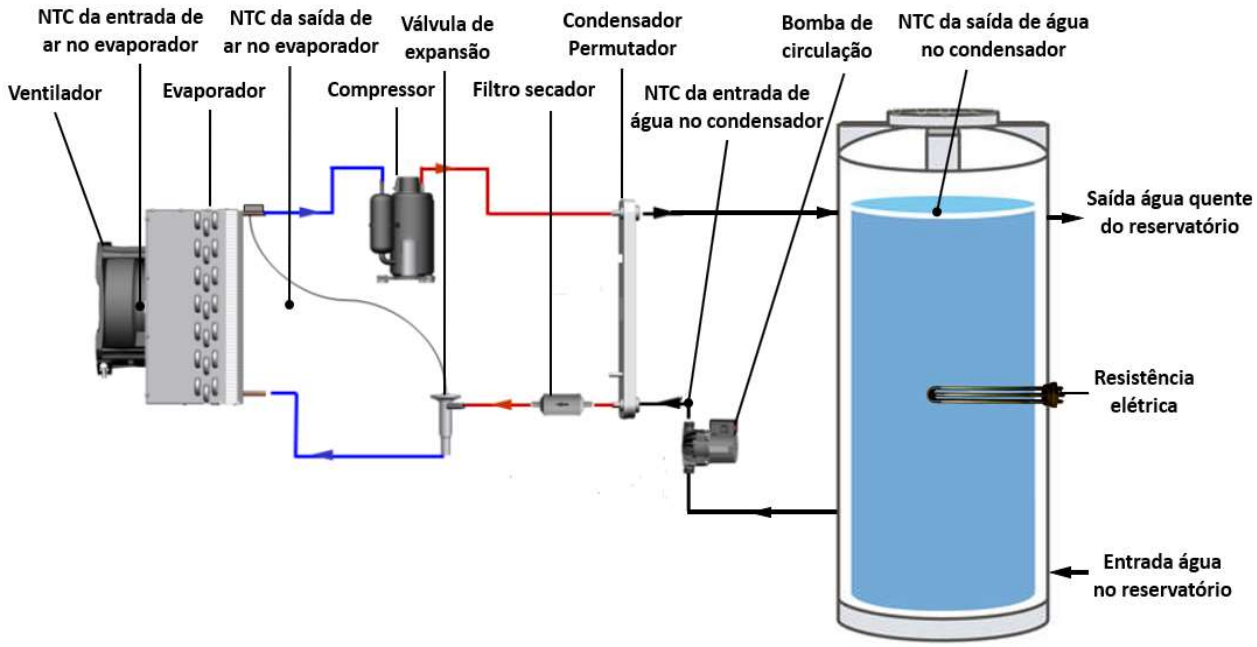


Figura 8: Esquema da disposição dos componentes da bomba de calor.

3.3. Problemas mais comuns de uma bombas de calor e respetivas causas

•Compressor

O compressor é onde existe mais movimentos mecânicos, podendo ocorrer o aumento da fricção se não houver uma lubrificação eficiente, sendo bastante prejudicial para o equipamento. Uma consequência deste problema pode ser o aumento da temperatura no compressor, que pode ser detetado. Outro fator que pode aumentar a temperatura no compressor é a falta de fluido operante no sistema.

Outros componentes importantes do compressor são os rolamentos, cujo mau funcionamento, ou por fadiga, falta de lubrificação ou pela presença de líquido no interior do compressor, provocará ruídos indesejados e vibrações anormais, podendo ser também causadas por falha nos fixadores estruturais do sistema.

•Condensador/Permutador de calor

O condensador é o dispositivo onde o refrigerante cede calor para o sistema que vai aquecer a água do reservatório. Se este sistema estiver obstruído ou sem fazer a correta transferência de energia, o condensador elevará a sua temperatura podendo danificar cronicamente o equipamento.

•Válvula de expansão

A falta de fluido operante pode provocar uma temperatura demasiado alta à saída da válvula de expansão para ser possível haver trocas de calor no evaporador. Também o mau funcionamento do condensador, por falta de ventilação, pode causar este excesso de temperatura.

•Evaporador

O evaporador é onde o fluido refrigerante evolui de mistura bifásica líquido mais vapor saturado para vapor ou mesmo vapor sobreaquecido. Se a circulação de ar for deficiente, por colmatação ou deficiente funcionamento do ventilador, o refrigerante não passará a vapor e não aquecerá como pretendido, e entrará no compressor a uma temperatura muito reduzida ou como mistura bifásica. É, então, essencial ter em conta a temperatura do refrigerante á saída do evaporador.

Capítulo IV

Hardware e *Software* implementado

Na componente prática desta dissertação foram implementados diversos dispositivos e desenvolvida programação recorrendo a algumas aplicações. Nesta secção são apresentados, em primeiro lugar, os equipamentos aplicados na bomba de calor para cumprir os objetivos propostos, e posteriormente o software utilizado tanto para realizar a comunicação entre dispositivos como para criar a interface com o utilizador. Para além disto é desenvolvido e explicado o algoritmo preditivo implementando-o no projeto.

A solução inicial para cumprir com os objetivos propostos na secção 1.2. é ilustrada na Figura 9. Esta apresenta a localização e o tipo de sensor necessário para obter as informações sobre o estado do sistema, quais os componentes onde foram instalados os interruptores necessários para controlar a bomba de calor e o fluxo de informação entre a interface com o utilizador, controlador do sistema, e os componentes. No decorrer deste documento são apresentadas as razões pela qual a solução assumiu esta arquitetura e quais as ferramentas utilizadas para a sua operação.

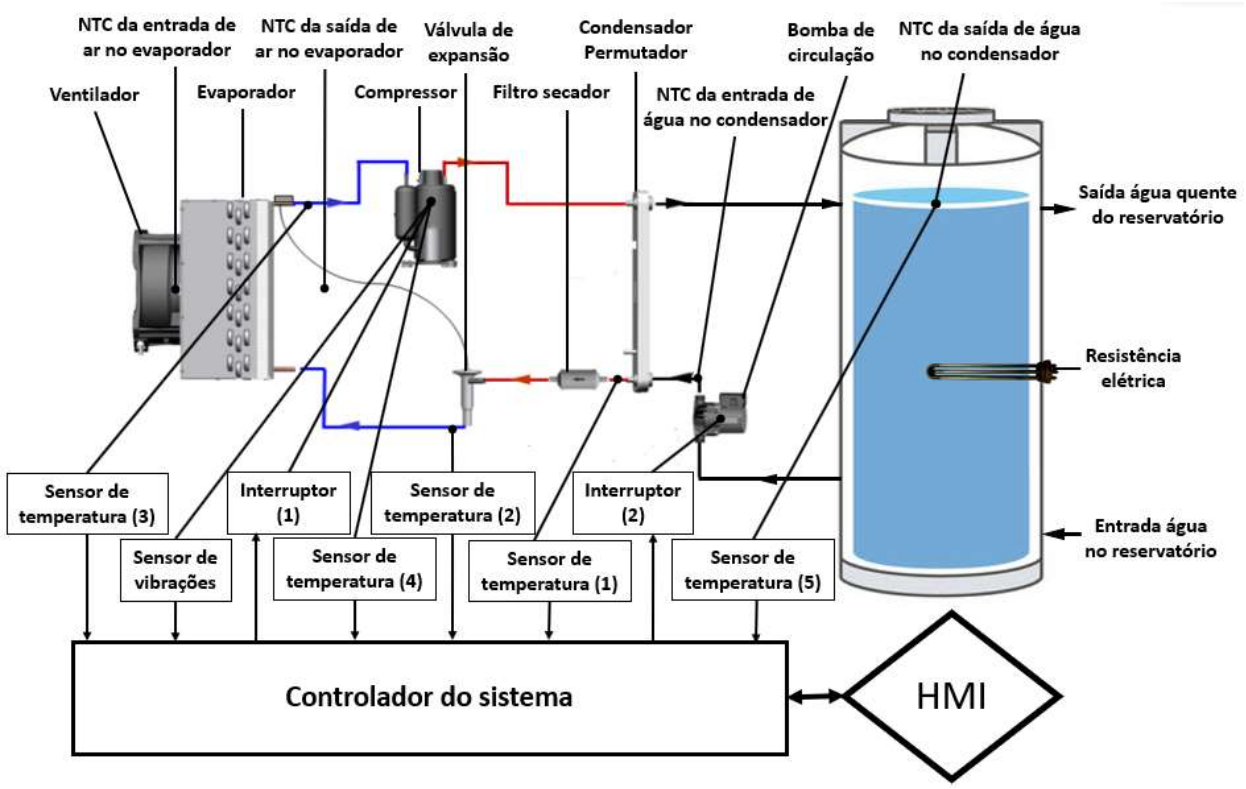


Figura 9: Esquema da disposição dos componentes da bomba de calor e dos componentes do sistema desenvolvido.

A Figura 9 pode ser descrita como um resumo inicial da solução do projeto, reunindo todos os dispositivos necessários para cumprir objetivos propostos. Observa-se nesta a necessidade de implementação de cinco sensores de temperatura, um sensor de vibrações, dois interruptores, um controlador do sistema e uma interface com o utilizador.

4.1. *Hardware*

O hardware são todos os elementos físicos utilizados no projeto para fazer a aquisição, o armazenamento e o tratamento dos dados. Faz parte da primeira fase do trabalho, pois é necessário escolher os sensores através dos quais se fará a aquisição dos valores, e o controlador do sistema, que armazenará e tratará os valores recebidos. A compatibilidade entre os sensores e os controladores é da maior importância, uma vez que cada sensor apresenta a sua forma de comunicar, ou por I2C ou por SPI. O controlador terá de suportar estes protocolos e ter portas de entrada suficientes para todos os dispositivos.

Nesta etapa começa-se por definir que valores retirar do sistema da bomba de calor e que sensores melhor cumprem esse propósito. De seguida foram analisados os sensores escolhidos, tanto a sua forma de comunicar como a preparação para dar início à aquisição dos dados. Por fim, foi feita uma análise aos microcontrolador e microcomputador a usar para que se possa apresentar um circuito elétrico completo e operacional.

4.1.1. Sensores

Antes de partir para a aquisição de dados é necessário definir que dados serão recolhidos e como serão obtidos. Para isto é imperativo fazer uma análise dos possíveis problemas que podem surgir, e escolher a melhor técnica para os identificar. Após esta análise foi possível retirar conclusões sobre a necessidade de quantificar as vibrações no compressor e monitorizar as temperaturas em vários pontos do circuito.

As vibrações provocadas pelo compressor dão indicações preciosas de possíveis falhas no mesmo. Como antes referido, as vibrações irregulares podem ser causadas por fixadores danificados, líquido refrigerante no interior do compressor ou rolamentos fatigados. Por isso, torna-se essencial acoplar um giroscópio à carcaça do compressor de maneira a que haja uma quantificação destes movimentos – sensor de vibrações da Figura 9.

As temperaturas registadas nos vários pontos de um ciclo termodinâmico podem ser bem definidas, e muitas vezes precisas para um funcionamento ótimo. Qualquer variação significativa da temperatura poderá indicar um possível mal funcionamento do sistema. Por isto, os sensores de temperatura são os mais utilizados neste projeto, sendo instalados em localizações estratégicas nos vários componentes. No compressor foi aplicado pois a falta de lubrificação ou fluido operante aumenta a sua temperatura – sensor de temperatura (4) da Figura 9. À saída do condensador/permutador foi instalado porque a temperatura pode subir em demasia devido à falta de transferência de calor para a água, devido a um entupimento, ao deficiente funcionamento da bomba hidráulica ou a falta de água no reservatório – sensor de temperatura (1) da Figura 9. À saída da

válvula de expansão, a falta de fluido operante provoca um aumento de temperatura, sendo muitas vezes fatal para o sistema – sensor de temperatura (2) da Figura 9. Foi aplicado também um sensor de temperatura à saída do evaporador, uma vez que o seu mau funcionamento provoca uma diminuição significativa da temperatura neste ponto – sensor de temperatura (3) da Figura 9. Para além destes indicadores de temperatura no circuito, foi aplicado um outro sensor de temperatura no reservatório, não só para informar o utilizador da temperatura da água como também para confirmar que objetivo da bomba de calor está a ser cumprido – sensor de temperatura (5) da Figura 9.

A seguir são especificados os sensores utilizados, o seu modo de funcionamento e quais os valores registados e enviados para o controlador do sistema.

- **GY-521 MPU 6050**

O sensor GY-521 MPU 6050 possui um poderoso chip da empresa InvenSense que proporciona uma solução com giroscópio de três eixos, acelerómetro de três eixos e um termómetro. Conta ainda com um processador digital de movimento que, com auxílio de um algoritmo, combina os vários sensores num sensor de 9 eixos. Utiliza uma comunicação por meio de barramento I2C, não necessita de intervenção do processador do sistema, e tem um baixo custo, tornando-se um dispositivo muito apelativo. Na Figura 10 é apresentado o sensor com a definição das suas ligações para comunicação, [13].

Figura 10: Sensor GY-521 MPU6050

Após as ligações do GY-521 serem efetuadas e o sensor ser "acordado", pondo a posição de memória *Power Management 1* com o valor 0x00, este inicia o registo de dados para as suas posições de memória. Estes registos são as quantificações da orientação e vibração do sensor. Os dados de cada eixo são guardados em duas posições de memória, o mais e o menos significativos ocupando um total de 16 bits. Cada posição de memória tem o seu endereço que será utilizado para selecionar os valores a ler. Na Tabela 2 são apresentadas as posições de memória utilizadas e os respetivos endereços no sistema hexadecimal, [14].

ACCEL_XOUT_H	ACCEL_XOUT_L	ACCEL_YOUT_H	ACCEL_YOUT_L	ACCEL_ZOUT_H
3B	3C	3D	3E	3F
ACCEL_ZOUT_L	**GYRO_XOUT_H**	**GYRO_XOUT_L**	**GYRO_YOUT_H**	**GYRO_YOUT_L**
40	41	42	43	44
GYRO_ZOUT_H	**GYRO_ZOUT_L**	**TEMP_OUT_H**	**TEMP_OUT_L**	**PWR_MGMT_1**
45	46	47	48	6B

Tabela 2: Posições de memória e respetivos endereços do sensor GY-521 MPU6050.

O valor da temperatura é o único que precisa de ser convertido para a escala Celsius através da seguinte equação fornecida pelo fabricante.

$$T(°C) = \frac{TEMP_{OUT}}{340} + 36{,}53$$

- T(°C) – Temperatura real, na escala Celsius
- $TEMP_{out}$ – Temperatura proveniente do sensor para conversão

O sensor GY-521 MPU 6050 vai ser aplicado no compressor e funciona como sensor de temperatura (4) e sensor de vibrações da Figura 9 na solução final do protótipo.

- **TC-74**

O sensor digital de temperatura TC74 é fornecido pela empresa Microchip a um custo de pouco mais de um euro a unidade. Devido ao seu tamanho reduzido torna-se muito útil em projetos onde o espaço é limitado. Este sensor pode ser usado em série, utilizando o protocolo I2C para comunicar. Possui uma resolução de 1°C, e pode atingir temperaturas desde -40°C até 125°C. Está preparado para recolher oito registos por segundo. Na Figura 11 é apresentado o sensor com a indicação dos seus pinos para comunicação.

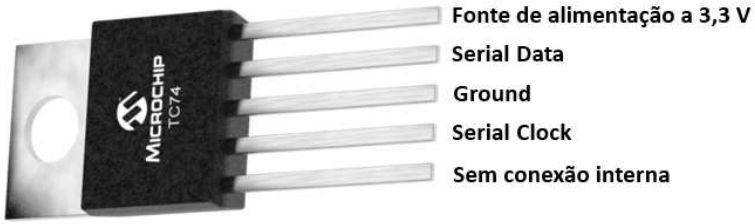


Figura 11: Sensor TC-74

Existem vários tipos deste sensor que diferem no formato, nos valores de tensão utilizados e nos endereços para comunicação. Este último limita a quantidade de sensores possíveis de juntar num barramento I2C, sendo no máximo oito elementos. Para o trabalho proposto foram apenas

utilizadas as versões A0 de 3,3 V e A5 de 5,0 V com a disposição semelhante à apresentada na Figura 10. Na Tabela 3 podem observar-se todos os endereços existentes neste modelo.

TC74-A0	TC74-A1	TC74-A2	TC74-A3	TC74-A4	TC74-A5	TC74-A6	TC74-A7
48	49	4A	4B	4C	4D	4E	4F

Tabela 3: Posições de memória do sensor TC74 e respetivos endereços, [15].

Os valores recebidos não necessitam de qualquer conversão, encontrando-se já na escala de Celsius.

Um pormenor destes sensores de temperatura é que têm a necessidade de pôr os condutores SDA e SCL à voltagem de 3,3 V conforme requisito do fabricante, pois a maneira de o *Master* comunicar é "puxar" a tensão a 0 V. Posto isto, foram escolhidas resistências de 10 KΩ para ligar o TC74-A0 e o TC74-A5 a 3,3 V, sendo os dois alimentados a 3,3 V.

No sensor GY-521 MPU 6050 tal não foi necessário, pois já possui este sistema internamente, sendo possível ligar diretamente os condutores de comunicação ao microcontrolador.

Os sensores TC-74 foram aplicados na solução final e assumiram as posições sensor de temperatura (1), sensor de temperatura (2) e sensor de temperatura (3) na Figura 9.

- **MAX6675 K**

O sensor MAX6675 K é produzido pela empresa Maxim Integrated Products, Inc., com o objetivo de quantificar a temperatura no seu terminal, transformando-a num sinal elétrico de tensão variável. Mais uma vez é um sensor com um custo inferior a dois euros, com condutores compridos que possibilitam a chegada do terminal a locais de difícil acesso. A incerteza das medições é de ±0,25°C, e pode medir temperaturas de 0°C a 1024°C. O protocolo de comunicação com que opera é o SPI e, por isso, conta com três condutores de comunicação, uma vez que não necessita do SDA_{in}. Está programado para enviar constantemente o valor correspondente à temperatura pelo seu SDA_{out}, e não sendo preciso estar o *Master* a pedir os valores torna-se dispensável a utilização do SDA_{in}, [6].

Na Figura 12 é apresentado o sensor com indicação dos terminais de alimentação e comunicação.

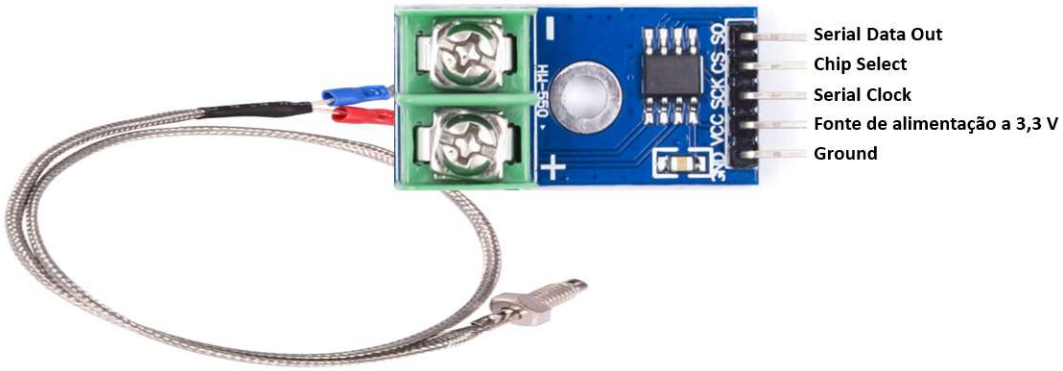


Figura 12: Sensor MAX6675 K

Este termopar juntamente com o conversor geram um sinal de 12 bits após a conversão de analógico para digital, e contando com alguns sistemas de correção de temperatura é possível obter valores fiáveis. Quando os valores chegam ao controlador do sistema, é necessário recorrer à equação fornecida pelo fabricante para obter a correta temperatura dos terminais:

$$T(°C) = T_{Amb} + \frac{V_{out}}{5 * 41\mu V/°C}$$

- T(°C) – Temperatura real
- T_{amb} – Temperatura ambiente
- V_{out} – Tensão proveniente do sensor para a conversão

É importante referir que num termopar do tipo K, como o que foi utilizado no projeto, a relação entre tensão e temperatura é de 41 µV/°C, ou seja, por 1°C a tensão no SDA_{out} varia 41 µV que posteriormente é convertido no MAX6675 para valores digitais e assim enviar por SPI.

O termopar MAX6675 K foi aplicado na localização sensor de temperatura (5) da Figura 9.

4.1.2. Interruptor *ON/OFF*

Apesar de os sensores serem muito importantes para a obtenção de dados do circuito da bomba de calor, é necessário atuar em conformidade após a análise dos valores recebidos. Essa ação passa por desligar a bomba de calor quando algum problema severo é detetado a partir dos registos, tornando essencial aplicar algum dispositivo no circuito que interrompa o funcionamento da bomba de calor e mantenha o equipamento e o operador seguros. Um modulo relé é o componente elétrico que melhor serve este propósito, pois consegue interromper correntes de 230v, sendo alimentado a 5 V pelo controlador do sistema.

- **Modulo Relé**

O modulo relé é um equipamento de simples manuseamento que serve para controlara fluxos de corrente elétrica de maior voltagem a partir das baixas tensões a que operam os controladores. A sua constituição conta com três componentes importantes que podem ser conferidos no esquema elétrico da Figura 13.

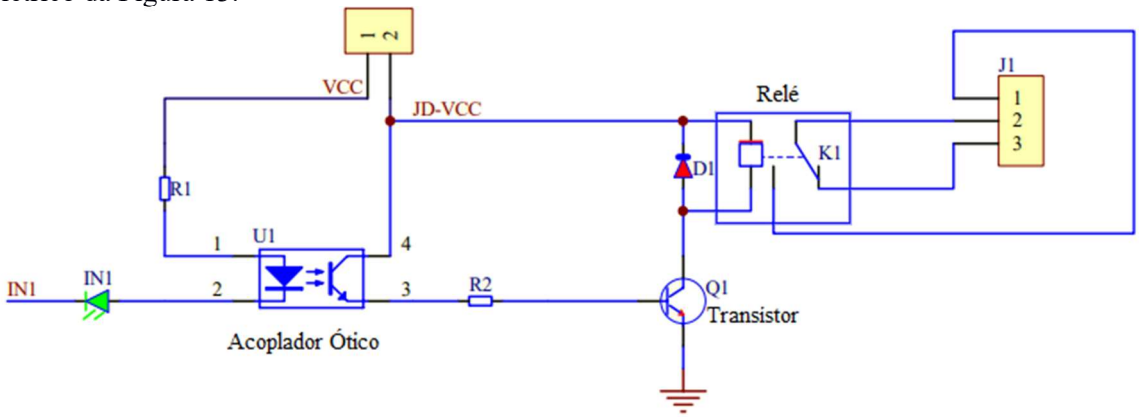


Figura 13: Esquema elétrico do Módulo Relé

O acoplador ótico é um componente de segurança que evita o contacto entre diferentes tensões pois transforma o sinal elétrico em luminuso por meio de um diodo emissor de luz e depois, a partir de um recetor, faz a passagem de corrente se este receber o sinal, garantindo assim que as tensões de 230 V nunca chegam ao controlador. Após esta operação o sinal chega ao transístor, onde decide se a corrente passa pelo relé. Se o relé for alimentado faz a troca de estado do seu interruptor interno e permite a passagem de corrente de 230 V, podendo ligar o equipamento em questão.

O modulo utilizado neste protótipo foi o 4 Relay Module 2PH63083A que tem a capacidade de controlar quatro circuitos elétricos. Com este equipamento seria possível controlar a bomba de calor a partir dos seus componentes internos, como o compressor, o ventilador ou a bomba hidráulica, substituindo por completo a HMI original. Apesar disso apenas foram ligados ao modulo relé o compressor e a bomba hidráulica o que tornou imprescindível a utilização da interface original da bomba de calor. Esta etapa do projeto foi assim simplificada pois se procedesse à completa substituição da HMI haveria a necessidade de programar cada interação de cada componente, levando bastante tempo e fugindo ao foco do projeto.

O dispositivo pode ser observado na Figura 14, tal como os seus *inputs*, *outputs* e as suas respetivas descrições.

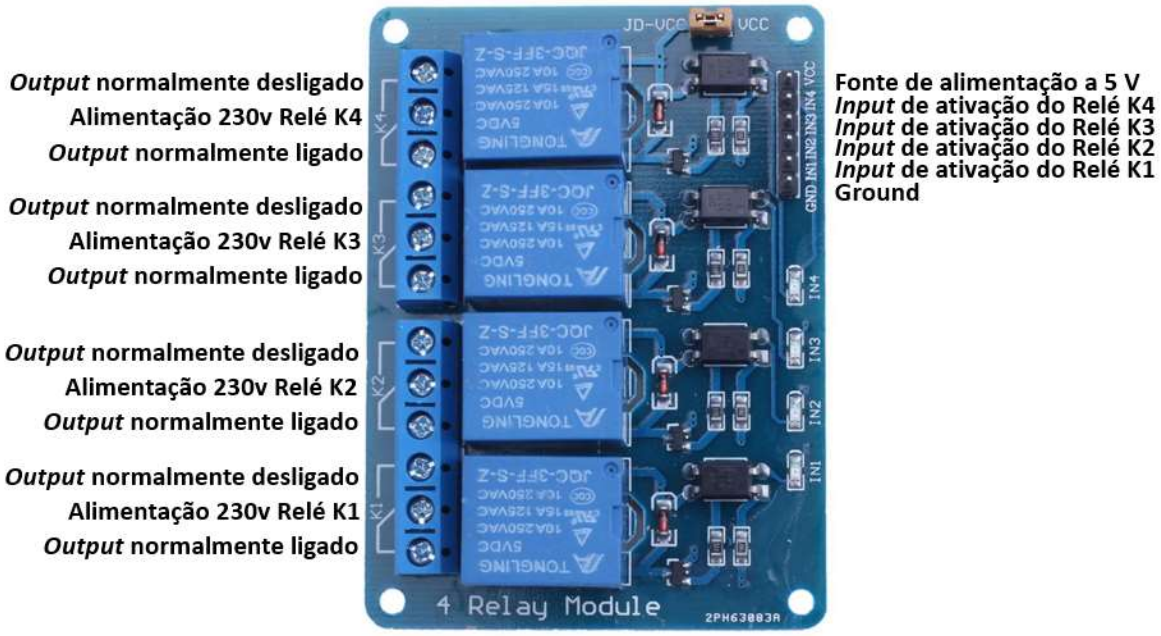


Figura 14: Modulo Relé

O modo de funcionamento baseia-se em manter a entrada de ativação a 0 V para acionar o *output* normalmente desligado e quando trocar para 3,3 V passa a ser ativado o *output* normalmente ligado. Esta placa tem de ser alimentada a 5 V para que assuma um correto funcionamento.

Este modulo relé assumiu o papel de interruptor (1) e interruptor (2) na Figura 9 controlando a bomba de circulação e o compressor de vapor.

Com este último dispositivo conclui-se a apresentação dos acessórios utilizados neste trabalho. É importante referir que existem no mercado outros equipamentos que cumprem com os mesmos objetivos, estando esta escolha dependente da disponibilidade dos dispositivos no laboratório de eletrónica. Segue-se agora a escolha do controlador mais indicado para o objetivo proposto, sendo necessário realizar uma análise em função das duas opções que foram abordadas no decorrer dos trabalhos.

4.1.3. Controlador do sistema

O controlador do sistema na Figura 9, como o próprio nome indica, vai gerir a comunicação entre sensores, realizando todas as operações lógicas necessárias para que exista uma base de dados com a evolução temporal das variáveis medidas em cada ponto do sistema. Para que isto seja possível recorreu-se a dois equipamentos com caraterísticas distintas, e avaliando cada um para que o controlador nem ficasse aquém nem muito além dos requisitos pretendidos.

Iniciou-se a análise pelo microcontrolador, e em seguida, pelo microcomputador, para finalizar com a melhor opção possível.

- **Microcontrolador ESP8566**

Um microcontrolador é um circuito integrado que realiza funções de lógica matemática, que permite o correto funcionamento do sistema de controlo. As caraterísticas principais são possuir muita autonomia operativa e ter um preço muito baixo. Para esta análise foi selecionado o microcontrolador ESP8266 pela sua simplicidade de operação, [16].

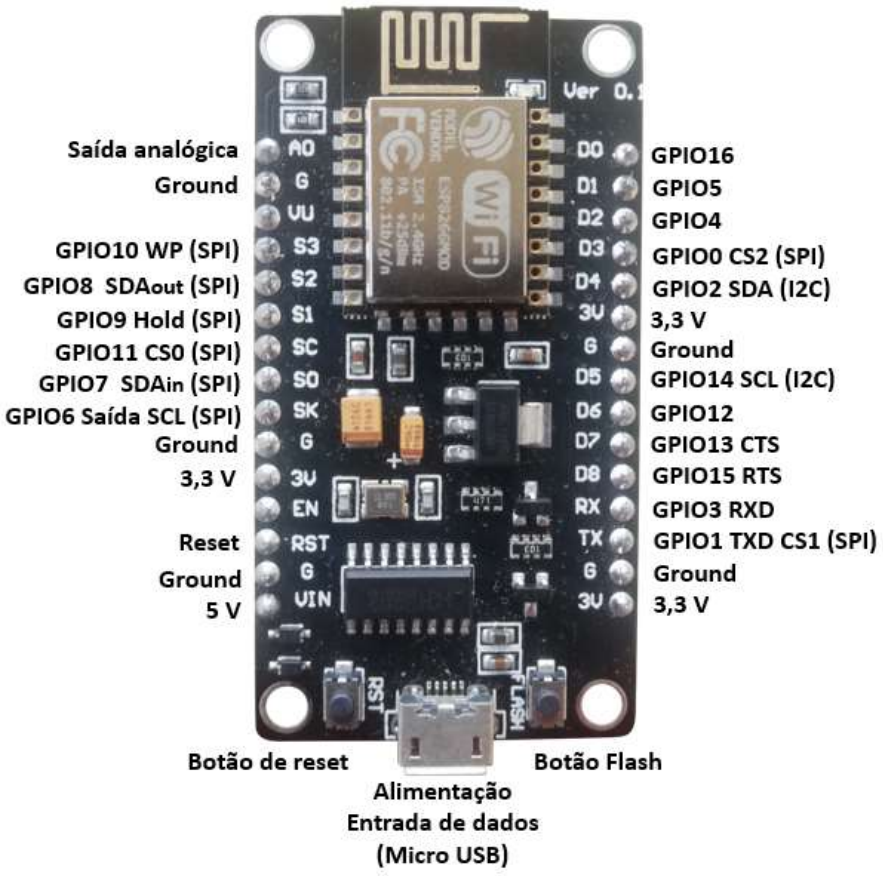


Figura 15: Microcontrolador ESP8266MOD.

O microcontrolador ESP8266, fabricado pela Espressif, suporta rede de *WiFi* completa de forma contínua e eficiente, tem um design compacto e é bastante fiável na indústria da *Internet of Things*. Pode ser adaptado a qualquer projeto como adaptador *WiFi* através da interface SPI e/ou I2C, dando a possibilidade de se conectar com diversos tipos de sensores. Na Figura 15 estão ilustradas as saídas do ESP8266 e a sua devida explicação.

O microcontrolador apresenta soluções diversificadas de comunicação através dos seus GPIO para que haja compatibilidade com a maior parte dos sensores do mercado, assegurando assim o transporte dos valores pretendidos para uma interface digital. Este procedimento será explicado mais adiante no capítulo relativo ao *Software*.

- **Microcomputador Raspberry pi 4, Model B**

A introdução dos microcomputadores marcou o início da revolução da sociedade tecnológica, pois a suas diversas aplicações chegam a qualquer parte do nosso quotidiano, podendo ser aplicados em contexto médico, controlo de processos, instrumentação, ou apenas para entretenimento doméstico. As suas características principais passam pelo seu design de tamanho reduzido, múltiplas portas de acesso, como USB, *Ethernet*, áudio Jack, HDMI e a sua capacidade de criar um ambiente de trabalho como qualquer computador. Esta última característica é muito importante para que haja a possibilidade de criar interfaces com o utilizador apelativas e programar no próprio aparelho, facilitando a etapa de testes. A utilização de uma memória ROM poupa problemas no armazenamento de dados, programas e servidores necessários para o cumprimento dos objetivos.

O Raspberry pi 4 é um microcomputador desenvolvido pela Raspberry Pi Foundation, que tem como principal objetivo incentivar a aprendizagem da informática básica. O dispositivo foi desenvolvido numa placa única com compatibilidade para dois ecrãs 4k, teclado, rato, *WiFi* e *Ethernet*. Pode ser usado como um *desktop*, controlador de robot, reprodutor de multimédia, centro de network, controlador de uma empresa, bem como em muitas outras aplicações. Para além disto, conta ainda com um conjunto de pinos GPIO que abre portas para a compatibilidade com comunicações I2C e/ou SPI, ou seja, é assegurada a comunicação com os sensores escolhidos previamente. Na figura 16 está representado o microcomputador Raspberry pi 4 com a indicação das portas de conectividade.

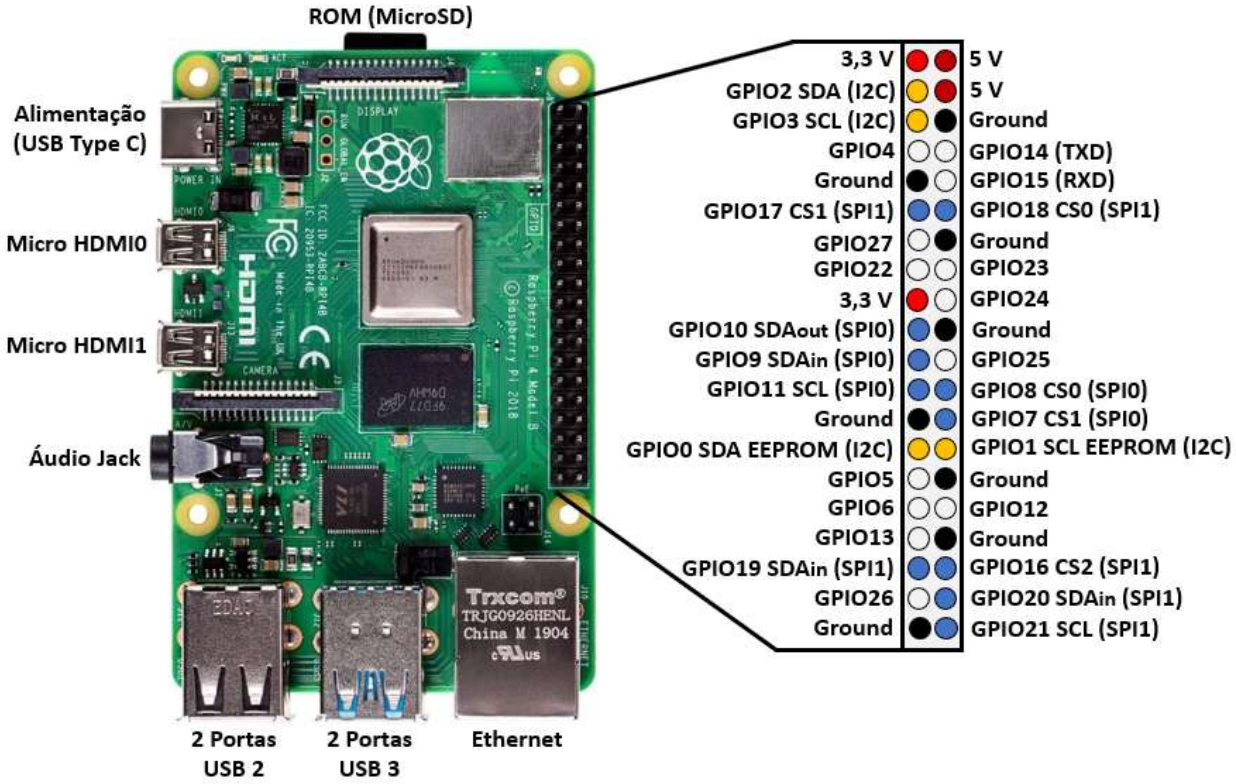


Figura 16: Microcomputador Raspberry pi 4, Model B.

- **Qual o controlador mais indicado?**

Após a análise das caraterísticas do microcontrolador, do microcomputador e dos requisitos do sistema em análise pode-se afirmar que é possível usar tanto um como o outro para que os objetivos do trabalho sejam alcançados. No entanto, o decorrer do trabalho não seria de todo idêntico, tal como as soluções finais seriam ligeiramente diferentes e, consequentemente, com vantagens e desvantagens distintas. Posto isto, no decorrer deste trabalho será apresentada em paralelo a solução tanto para o ESP8266 como para o Raspberry pi 4, uma vez que ambas apresentam características relevantes para o utilizador e para o fabricante da bomba de calor.

4.1.4. Circuitos elétricos

O circuito elétrico foi elaborado de forma idêntica para ambos os controladores vistos que os sensores podem ser utilizados da mesma forma para cada um dado ambos possuírem interfaces de comunicação I2C e SPI. Os sensores utilizados foram três TC74, um GY-521 MPU 6050 e um MAX6675 K, ligados aos pinos GPIO através de três barramentos, dois I2C e um SPI. O motivo da criação de dois barramentos I2C prende-se com o facto de apenas ter disponíveis dois modelos de sensor TC74, o A0 e o A5, sendo que houve a necessidade de usar dois TC74-A0. Ao usar dois modelos iguais os seus endereços seriam os mesmos e não haveria possibilidade de distinguir qual deles estaria a responder ao *Master*. Então, foi colocado um em cada barramento, garantindo a entrada dos valores nos controladores por portas distintas.

Em primeiro lugar foram feitas as ligações dos sensores ao microcontrolador de acordo com os respetivos *datasheets*. Preferencialmente, foram usados os pinos destinados aos protocolos de comunicação e, posteriormente, através do *software*, criada uma nova entrada I2C. Na figura 17 é apresentado o esquema com todas as ligações ao microcontrolador ESP8266.

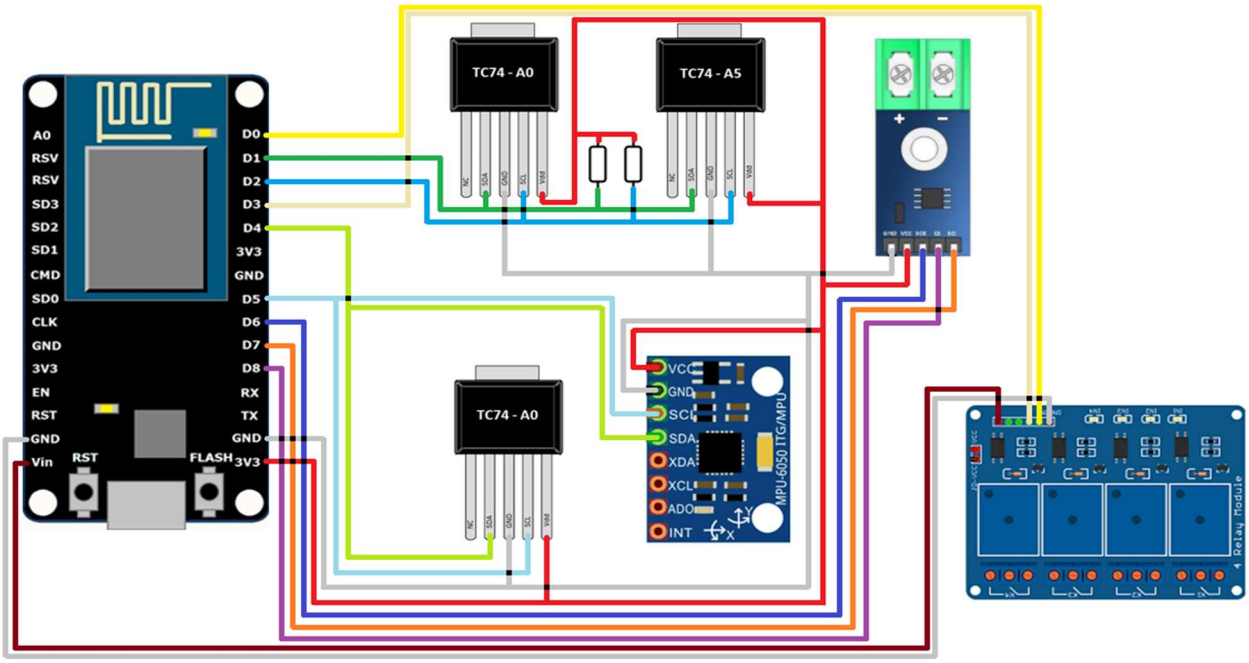


Figura 17: Esquema elétrico dos sensores e modulo relé utilizados com o microcontrolador ESP8266

Como referido anteriormente, os sensores TC74 necessitam de ter os condutores SDA e SCL a 3,3 V para que a comunicação seja possível; daí terem sido colocadas resistências a conectar estes condutores à alimentação dos sensores. Para o barramento do acelerómetro não foi necessário fazer esta conexão, uma vez que o sensor GY-521 MPU 6050 já apresenta este sistema internamente. No circuito apresentado os sensores são todos alimentados a 3,3 V. Já o modulo relé é alimentado a 5 V e acionado pela saída D0 e D3 do microcontrolador.

Em relação ao circuito do Raspberry pi foram utilizados essencialmente os mesmos princípios e ligações, diferenciando apenas os pinos GPIO. Este equipamento conta já com mais que um canal I2C e SPI, devido ao seu elevado número de pinos. No entanto, apenas um canal I2C está disponível para a conexão a sensores deste tipo uma vez que o segundo conjunto é dedicado apenas a EEPROM's. Por isso, teve de ser criado um novo canal para o segundo barramento em pinos de usos geral, neste caso, nos pinos GPIO23 e GPIO24 como se pode observar na Figura 18.

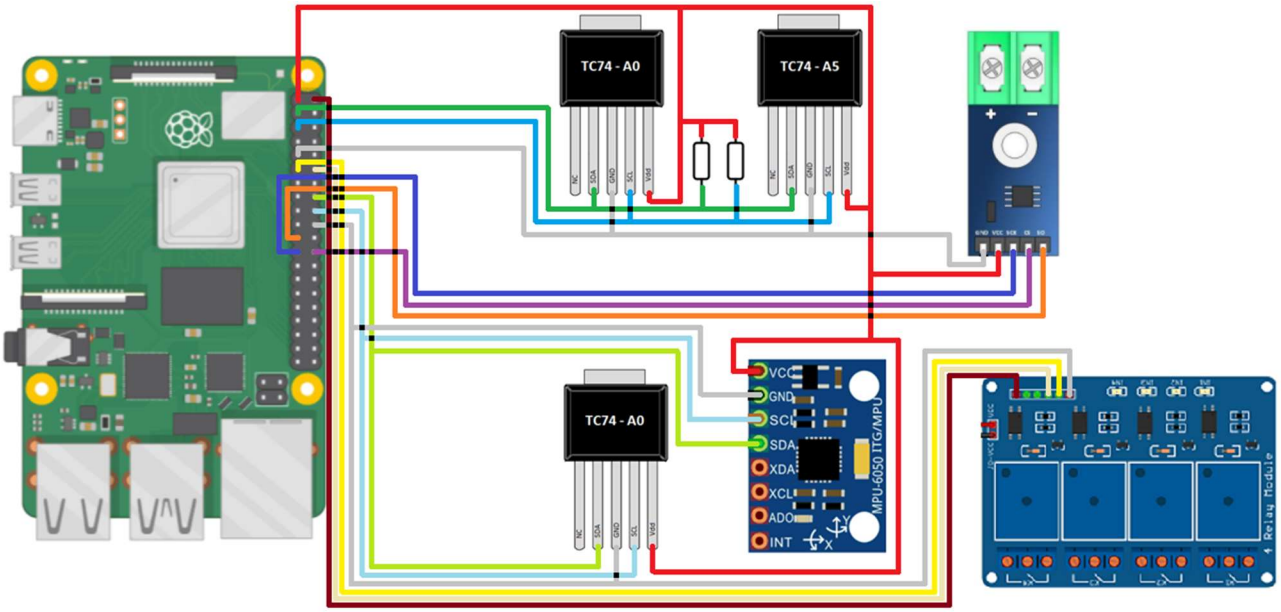


Figura 18: Esquema elétrico dos sensores e modulo relé utilizados com o microcomputador Raspberry pi 4.

As ligações foram feitas através dos fios de cobre e as conexões aos sensores com barramentos de conectores e cabos *Dupont*, para que fosse possível alterar facilmente os sensores e as suas conexões em caso de necessidade. Foram utilizadas também mangas térmicas para revestir alguns condutores que foram, inevitavelmente, soldados a estanho. Este procedimento preveniu o risco de eventuais curto-circuitos devido ao provável contacto de condutores se estes tivessem soltos e desprotegidos.

Nesta fase do desenvolvimento, a aplicação dos sensores de temperatura na bomba de calor foi feita através de fita adesiva PVC impermeável, de modo a obter a melhor condutividade térmica possível. O controlador foi alimentado por uma fonte externa e sem qualquer conexão ao HMI da bomba de calor.

4.2. Software

O software é a componente lógica do sistema, define os parâmetros de obtenção, tratamento e disponibilização dos dados para o utilizador. Para isso, foram usadas ferramentas informáticas dedicadas a cada tarefa e desafio que surgiram durante o trabalho. A partir destas aplicações, e recorrendo a bibliotecas e funções, foram surgindo as soluções que cumprem os objetivos desta dissertação. De seguida são abordados os ambientes de desenvolvimento e as bibliotecas usadas para cada um dos controladores, juntamente com a devida explicação. Depois da obtenção dos dados ficaram explicitas quais as técnicas para a sua disponibilização e qual o programa para a obtenção da HMI deste sistema, primeiro para o microcontrolador e depois para o microcomputador. Por fim são detalhadas quais as operações de previsão e prevenção realizadas a partir da base de dados criada.

4.2.1. Arduino no microcontrolador ESP8266

A programação do microcontrolador teve lugar no software Arduino, que é um ambiente de desenvolvimento flexível e fácil de operar. O código, em linguagem de programação C#, é organizado por funções, sendo que a função *main()* é criada assim que se abre um novo projeto por *default* e o programador apenas trabalha nas funções *setup()* e *loop()* que a constituem. Na primeira função estão as configurações do programa, como a declaração de variáveis e entradas e saídas do microcontrolador, uma vez que este é executado apenas quando se inicia o ESP. A segunda função está contida dentro de um ciclo *while* e está sempre a ser executada, sendo possível receber dados de sensores e enviá-los para a *Web* constantemente. Para além do ambiente de desenvolvimento, o Arduíno tem também um monitor série que permite saber o que se está a passar no microcontrolador, constituindo assim um poderoso sistema de *debug*. Na Figura 19 é apresentada a interface Arduíno, [2].

Figura 19: Ambiente de desenvolvimento Arduino

O Arduino possibilita a inclusão de bibliotecas previamente concebidas para que seja facilitada a implementação de vários comandos essenciais à comunicação com os sensores e com a *Internet* por *WiFi*. Neste trabalho foram incluídas algumas bibliotecas que serão abordadas de seguida, mostrando quais as funções utilizadas e o seu propósito no programa.

- **Biblioteca *"Wire"***

A biblioteca *"Wire"* permite que o microcontrolador comunique pelo protocolo I2C com dispositivos diversificados, que também possuam esta capacidade, como acelerómetros, EEPROM's e conversores analógico-digital, entre outros. A biblioteca tem uma função para cada evento, necessário para a comunicação I2C anteriormente analisada. Na Tabela 4 estão indicadas as funções mais importantes que foram utilizadas neste trabalho, como a sua explicação, [17].

Função	Descrição
Wire.begin(D4, D5)	Ativa a biblioteca e define os pinos do ESP que serão utilizados para ligar os condutores SDA, D4 e SCL, D5.
Wire.beginTransmission(0x68)	Inicia o envio da mensagem pelo *Master* ao executar o evento I. Depois, envia o endereço mais o bit de escrita (0) para que o *Slave* correspondente seja acionado, esperando o evento ACK como resposta.
Wire.write(0x3B)	É a função que envia dados para o *Slave*. Envia oito bits (um byte) e espera como resposta o evento ACK. Serve para selecionar registos de memória e enviar dados para esses registos.
Wire.endTransmission()	Dá por terminada a mensagem, gerando o evento F.
Wire.requestFrom(0x68, 14)	Serve para ler valores de um *Slave*. O *Master* começa por gerar um evento I e envia o endereço do *Slave* que quer ler mais o bit de leitura (1). Espera a resposta com o ACK e com os dados pedidos. Depois, envia o evento NACK e F para finalizar a mensagem. Repete-se até obter o número de valores pedido, 14.
Wire.read()	Serve para ler os bytes enviados pelo *Slave* para que possam ser armazenados numa variável.
Wire.available()	Informa se a conexão está estabelecida entre o *Master* e o *Slave*.

Tabela 4: Funções da biblioteca Wire e respetiva descrição.

As funções foram usadas de forma a obter os valores dos sensores TC-74 e GY-521 MPU 6050 e armazená-los em variáveis para o seu processamento. Os sensores de temperatura não necessitam de qualquer intervenção, uma vez que são recebidos valores de 8 bits e convertidos diretamente, estando prontos para serem enviados para a interface HMI.

Relativamente ao sensor GY-521 MPU 6050 foram guardados os dados do acelerómetro e do giroscópio dos 3 eixos e, adicionalmente, a temperatura. Estes valores foram guardados em variáveis de 16 bits, pois são constituídos cada um por dois bytes, o mais e o menos significativo. Uma vez que a comunicação I2C apenas envia 8 bits por mensagem, foi necessário adquirir o primeiro byte, movê-lo oito bits para a esquerda e voltar a adquirir novo byte, preenchendo os 16 bits da variável. Os valores registados pelo acelerómetro não foram utilizados, pois apenas serviriam para indicar a posição do sensor, que não é interessante para o objetivo final. Posto isto, com os valores do giroscópio foi estimado um limite máximo e criada uma variável booleana que passava a 1 sempre que um dos eixos infringisse esse limite, sendo mais fácil enviar a informação relativa às vibrações do compressor.

Em relação à temperatura proveniente do sensor GY-521 MPU 6050, como já referido na secção 4.1.1., foi preciso utilizar a equação fornecida pelo fabricante para a sua conversão, sendo utilizada a mesma lógica dos sensores TC-74.

- **Biblioteca *"max6675"***

A biblioteca *"max6675"* tem ferramentas dedicadas para a comunicação com o sensor MAX6675, executando as operações necessárias do protocolo SPI. A sua utilização permitiu obter facilmente os valores da temperatura dos termopares do tipo K sem que houvesse a necessidade de operações suplementares. As funções aplicadas estão explicadas na Tabela 5.

Função	Descrição
MAX6675 thermocouple(D6, D8, D7)	Define quais as saídas GPIO usadas para a comunicação SPI do sensor, onde D6 é o SCL, D8 é o CS e D7 é o SDA_{out}.
thermocouple.readCelsius()	Funciona como leitor e conversor da tensão do condutor SDA_{out} e retoma o valor da temperatura no terminal.

Tabela 5: Funções da biblioteca max6675 e respetiva descrição.

- **Biblioteca *"math"***

No programa houve a necessidade de calcular variáveis e, consequentemente, fazer operações de arredondamentos para que não fossem enviadas para a base de dados variáveis com muitas casas decimais. Isto apenas foi possível recorrendo à biblioteca *"math"* que conta com inúmeras funções que fazem as mais diversas operações matemáticas. No código apenas foi usada a função *round()*, que arredonda os valores à unidade.

- **Biblioteca *"esp8266wifi"***

Nesta fase os dados dos sensores já estão recolhidos e prontos para serem enviados para uma plataforma mais apelativa, onde possa haver interação com o utilizador e seja possível processar ações concretas de intervenção no funcionamento da bomba de calor, se assim for necessário. Para isso é necessário conectar o sistema com a *Internet*, instalando uma biblioteca no Arduino chamada *"esp6266wifi"*, que permite a ligação de uma forma simples e rápida. A biblioteca é constituída por diversas funções, mas apenas serão explicadas na Tabela 6 as que foram utilizadas no código do programa desenvolvido, contribuindo assim para a objetividade deste documento.

Função	Descrição
WiFiClient TCP_Client	Cria o objeto do tipo TCP/IP para que o ESP atue como servidor.
WiFi.mode(WIFI_STA)	Define o modo como o ESP vai funcionar, podendo ser no modo *Station* que assume um papel de dispositivo normal que se conecta à rede ou no modo *Access point* permitindo que outros dispositivos se conectem ao ESP e por sua vez à rede.
WiFi.begin(ssid, pass)	Inicia a ligação à rede conectando-se ao router. Para isto é necessário fornecer à função o nome da rede, ssid, e a palavra-chave, pass.
WiFi.status()	Função binária que indica se a ligação está estabelecida ou não.
WiFi.localIP()	Indica o endereço IP que que foi atribuído ao ESP quando se conectou á rede WiFi.

Tabela 6: Funções da biblioteca esp8266wifi e respetiva descrição

- **Biblioteca *"PubSubclient"***

A disponibilização das variáveis foi feita recorrendo ao protocolo MQTT, onde se recorreu a um *broker online* para publicar e subscrever toda a informação recolhida. Este *broker* pode ser encontrado no site "www.dioty.co", e apenas fazendo um *signin* com um email é possível criar um *host* onde são publicados os dados. No entanto, a versão gratuita tem um limite mensal de mensagens, mas que foi o suficiente para os testes realizados.

A comunicação do sistema com o *broker* foi estabelecida através do microcontrolador ESP8266, onde foi necessário instalar mais uma biblioteca que facilita os comandos de publicação e subscrição de dados. A biblioteca *"PubSubclient"* é dedicada à comunicação pelo protocolo MQTT, e tem diversas funções com propósitos únicos que estão apresentadas na seguinte Tabela 7.

Função	Descrição
client.setServer(Host, Port)	Define a que *broker* o ESP se vai ligar, indicando o endereço, *Host*, que, neste caso, é o mqtt.dioty.co, e a porta serial virtual que, neste caso, é 1883, por onde se dá a troca de informação.
client.connect("Nome")	Atribui o nome ao cliente ESP no *broker* sendo mais fácil identificar as publicações feitas por ele.
client.setCallback(callback)	Definir a função *callback*, que é chamada sempre que houver alguma publicação nova no *broker* que não tenha sido feita pelo ESP, nos *topic* indicados pela função seguinte.
client.subscribe("topic")	Indica os *topic* que são subscritos pelo ESP no *broke*, e sempre que o valor destes for alterado no *broker* é executada a função *callback* no ESP.
client.connected()	Verifica se o ESP está conectado ao *broker*, para que se possa prosseguir a publicação de mensagens.
client.loop()	Faz a atualização da conexão e processa a mensagem recebida.
client.publish("topic", "msg")	Publica a mensagem no *broker* com um *topic* para ser trabalhada posteriormente.

Tabela 7: Funções da biblioteca PubSubclient e respetiva descrição.

Nesta etapa do desenvolvimento os sensores já estão a enviar dados para o microcontrolador e, consequentemente, a serem disponibilizados num servidor online, estando completo o objetivo do trabalho que consiste na monitorização do sistema da bomba de calor a partir de qualquer parte do Mundo com acesso à *Internet*. O próximo passo consiste em criar uma interface para que o utilizador tenha acesso aos dados e que possa intervir no funcionamento da bomba de calor. Esta interface terá acesso ao *broker online*, de onde retirará diretamente os valores, e por isso todos têm de vir previamente processados de modo a que a HMI apenas os apresente. O processamento tem lugar no Arduino, onde foram definidos limites de temperaturas e vibrações.

4.2.2. *Python* no Raspberry pi 4

A programação do microcomputador foi feita em *Python* no próprio dispositivo, uma vez que este suporta um ambiente de trabalho e ferramentas de programação quando se instala o sistema operativo, Raspbian. O OS é desenvolvido em Linux e fornecido gratuitamente pelo fabricante, tornando bastante apelativa a sua utilização. Recorrendo a um ecrã foram feitas as configurações iniciais e ativadas as comunicações I2C, SPI e VNC - Virtual Network Computing. Esta última serve para ter acesso ao ambiente de trabalho do Raspberry a partir de um qualquer computador que esteja conectado à mesma rede *WiFi* e que tenha instalado o software *VNC Viewer*. Isto facilitou a programação do microcomputador, pois não foi preciso teclado, rato ou ecrã suplementar para a elaboração do código.

O código foi escrito em linguagem *Python*, e teve o mesmo princípio que o do microcontrolador onde foi criada uma função com as configurações iniciais dos sensores e servidores para a correta comunicação. De seguida foi utilizado um ciclo *while* infinito que recebe os dados e os envia repetidamente. Mais uma vez foi necessário recorrer a bibliotecas para que fosse possível estabelecer as ligações de todo o sistema, sendo a seguir indicadas e explicadas quais as funções usadas.

- **Biblioteca *"SMBus"***

O *System Management Bus* é uma forma de gerir comunicações e é adotada por bastantes circuitos como *motherbords*, pois facilita a implementação do protocolo I2C. Recorrendo à biblioteca *"SMBus"* foi dado acesso a funções que retiram diretamente os valores dos sensores apenas com uma linha de código, contribuindo para uma programação sucinta. Na Tabela 8 é explicado o funcionamento de cada função utilizada no projeto.

Função	Descrição
SMBus(1)	Define qual a saída GPIO utilizada para um dos barramentos previamente configurados no cmd do Raspberry pi. Neste caso o barramento 1 está nos GPIO 2 e 3.
i2cbus.write_byte_data (0x68, 0x6B, 0x00)	Escreve o valor 0x00 na posição de memória 0x6B do equipamento com endereço 0x68 que está no barramento i2cbus. Esta função faz todas as operações necessárias para enviar o valor pretendido pelo protocolo I2C.
i2cbus.read_byte_data (0x68, 0x3B)	Pede o valor da posição de memória 0x3B do equipamento com o endereço 0x68 que está no barramento i2cbus. Esta função faz todas as operações necessárias para obter o valor pretendido pelo protocolo I2C.

Tabela 8: Funções da biblioteca SMBus e respetiva descrição.

À semelhança do código do Arduino, foram adquiridos os valores do sensor TC-74 de forma direta, sendo que os dados do GY-521 MPU 6050 tiveram de ser adquiridos de 8 em 8 bits, perfazendo variáveis de 2 bytes. A temperatura deste último sensor foi sujeita à conversão fornecida pelo fabricante, como já explicado na secção 4.2.1.

- **Biblioteca *"spidev"***

A biblioteca *"spidev"* foi criada com o intuito de satisfazer as necessidades do protocolo SPI para uma correta comunicação a partir de um OS Linux. Em contraste com o Arduino, não é dedicada ao MAX6675, mas não deixa de ser simples e sucinta nas funções que usa para que os eventos do protocolo sejam cumpridos. Na Tabela 9 são abordadas essas funções.

Função	Descrição
spidev.SpiDev(0, 0)	Indica qual o canal SPI do Raspberry pi que vai ser usado e qual o CS desse canal. Neste caso foi usado o canal 0 e o CS0.
spi.readbytes(2)	Lê os bytes no condutor SDA_{out} que, neste exemplo, seriam 2 bytes, ou seja, 16 bits.

Tabela 9: Funções da biblioteca spidev e respetiva descrição.

Recorrendo ao *datasheet* do MAX6675 constata-se que os dados são enviados em duas remessas, 5 bits menos significativos - LSB e 7 bits mais significativos - MSB sendo que o resto dos bits são apenas de controlo e, por isso, têm de ser eliminados da variável. Assim, é necessário processar esta variável para que dos 16 bits sobrem apenas os 12 que são o valor da temperatura no terminal. Na Figura 20 é apresentada a mensagem SPI como chega do MAX6675 e após o seu processamento.

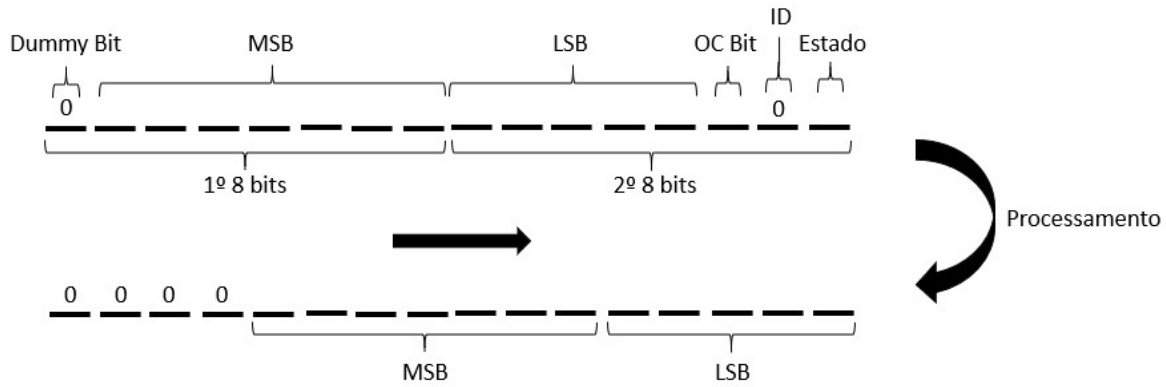


Figura 20: Processamento da mensagem SPI proveniente do MAX6675

A mensagem antes do processamento tem informações importantes, sendo o primeiro bit de teste e os 12 bits que o seguem a temperatura do terminal, sobrando três bits que são importantes de referir. O OC bit informa se o termopar está aberto ou fechado, ou seja, se os condutores do terminal não

estiverem em contacto os valores recebidos não variaram muito, mas não deixam de estar errados. A função deste bit é informar o estado do terminal, sendo que em condições normais é mantido a 0. O bit ID é sempre 0 para que seja fornecida uma identidade ao sensor, e o bit Estado pode assumir três estados de tensão. Este último serve para que seja possível partilhar o condutor SDA_{out} com mais dispositivos.

- **Biblioteca *"time"***

Os sensores estão preparados para enviar mensagens a grandes velocidades, e muitas das vezes é necessário limitar a cadência para que não haja colisões. Isto foi conseguido através de um timer que funciona como uma pausa no código disponível na biblioteca *"time"*. A função utilizada foi *time.sleep()*, onde se informa quantos segundos se pretende pausar a execução do programa limitando a cadência de dados por minuto, para que não haja uma base de dados exageradamente grande com informação de pouca, ou mesmo nenhuma, utilidade.

- **Biblioteca *"paho.mqtt.publish"***

À semelhança do ESP8266, a disponibilização dos dados para a *Web* foi feita através do protocolo de comunicação MQTT, diferenciando apenas o local do servidor que, sendo um microcomputador, tem capacidades de criar o seu próprio *broker*. Isto faz com que haja exclusividade no acesso e que não exista limite de mensagens, contribuindo para um projeto mais independente de servidores externos que, muitas vezes, têm de ser pagos para satisfazer a necessidades. Para obter este *broker* particular foi instalado o Eclipse Mosquitto gratuitamente, que cumpriu o objetivo para ele estipulado. Após este passo, já se pode começar a publicar os dados dos sensores no servidor, sendo necessário instalar a biblioteca *"paho.mqtt.publish"* que fornece funções simples e de fácil compreensão. Na Tabela 10 estão indicadas as funções utilizadas como a respetiva descrição.

Função	Descrição
mqtt.Client()	Cria um objeto cliente para que as outras funções saibam a que objeto vão recorrer para fazer as operações necessárias.
client.connect("192.168.43.182")	Usa o objeto *client* criado previamente para o conectar ao broker, sendo fornecido o endereço do servidor que será usado.
client.publish("Tc","55")	Responsável pela publicação da mensagem no *broker*. Cria um *topic*, "Tc" com o respetivo *payload*, "55", onde estará o valor do sensor naquele momento.

Tabela 10: Funções da biblioteca "paho.mqtt.publish" e respetiva descrição

4.2.3. Human Machine Interface

A HMI - Human Machine Interface da Figura 9 é a interface que está entre o utilizador e a máquina, sendo o único contacto que poderá ser estabelecido com o equipamento através do software instalado e, por isso, não é só importante a interação e dinamismo da aplicação como também toda a parte do design e apresentação dos dados. Nesta dissertação, como estão a ser abordados dois equipamentos de controlo distintos, foi preciso criar duas interfaces, utilizando diversos recursos, para a programação e disponibilização dos valores dos sensores. A diferença mais significativa é entre os equipamentos, pois os locais de armazenamento das variáveis e o processamento do código são distintos, mantendo-se a disposição visual e a ferramenta de programação.

No decorrer deste capítulo são abordadas as ferramentas imprescindíveis ao desenvolvimento do sistema, não só para a obtenção da área de interação, como também para a possibilidade de enviar alertas e notificações quando o utilizador está com a interface em segundo plano, permitindo a rápida intervenção se o problema não for grave a ponto de provocar a interrupção automática do funcionamento da bomba de calor. O node-red foi a base para as HMI, sendo explicado de seguida qual o seu propósito, o seu modo de funcionamento e a programação.

- **Node-Red**

O Node-Red é uma ferramenta de programação que serve para conectar dispositivos físicos com serviços *online* de forma inovadora e intuitiva. Fornece um editor baseado em *nodes*, que pode ser utilizado enquanto o código está a correr. Este editor de código é construído em Node.js ganhando a possibilidade de correr em *hardwares* de baixo custo bem como na *Cloud*. Os nodes estão disponíveis em paletes e podem assumir variadíssimas funções. Uma dessas paletes é o *Node-red-dashboard*, que é necessário instalar no Node-Red, servindo para criar rapidamente um painel com dados interativos e apelativos. Este painel está organizado por grupos que contêm retângulos com largura padrão, mas com a possibilidade de personalizar a dimensão dos objetos. Conta com várias possibilidades de objetos como botões, gráficos, saídas de áudio, notificações e *sliders*, entre outros. Por fim, é uma ferramenta muito fácil de usar e com grande potencial.

Para além do *dashboard* foram instaladas outras paletes com funções essenciais para a conexão com o *broker*, para a interação com a base de dados abordada no seguimento do documento e para o envio de emails a reportar o estado da bomba de calor. Estes *nodes* estão apresentados na Tabela 11 como a sua descrição.

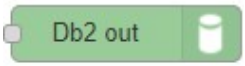
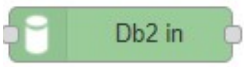
Node	Descrição
	Serve para enviar um número, uma *string* ou mesmo um objeto através de estimulações manuais ou automáticas de tempo a tempo. Foi usada para efeitos de *debug* quando não havia dados recebidos, e para fazer avaliações periódicas no programa final.
	Como o nome indica é um node de *debug*, que mostra numa janela o que está a receber, podendo identificar onde está o erro no fluxo.
	Recebe um valor, processa-o e envia o resultado. Foi o *node* mais utilizado no código, pois foi onde houve todo o processamento da informação e a escrita das notificações, dos emails, das mensagens para a base de dados e de toda a informação que aparece na HMI.
	Recebe uma mensagem e, de acordo com o seu valor, envia-o para um de dois caminhos no fluxo, funcionando como uma bifurcação.
	Utiliza o protocolo MQTT para comunicar com o *broker* e serve para receber mensagens, ou seja, é apenas para subscrever os valores quando estes são alterados, enviando-os através do fluxo.
	Utiliza o protocolo MQTT para comunicar com o *broker* e serve para enviar mensagens, ou seja, é apenas para publicar as variáveis que o utilizador pretende enviar para o controlador.
	Após o processamento da informação é-lhe fornecida uma mensagem onde o *topic* é o assunto e o *payload* é o conteúdo do email que será enviado para uma conta previamente definida.
	Serve para fazer a conexão à base de dados MySQL usada para o Raspberry, enviar e pedir dados específicos para o algoritmo de prevenção de falhas, abordado mais à frente.
	Serve para fazer a conexão à base de dados IBM Db2 on Cloud usada para o microcontrolador ESP8266 e enviar dados específicos para o algoritmo de prevenção de falhas abordado mais à frente.
	Serve para fazer a conexão à base de dados IBM Db2 on Cloud usada para o microcontrolador ESP8266 e pedir dados específicos para o algoritmo de prevenção de falhas abordado mais à frente.
	Controla qualquer pino GPIO digital do Raspberry, podendo acionar a saída com 3,3 V ou desativando com 0 V.

Tabela 11: Nodes utilizados para processar os dados no Node-Red e respetiva descrição.

Após todo o processamento dos dados e o seu devido armazenamento podem começar a ser apresentados ao utilizador a partir do *dashboard*. Este organiza toda a informação por grupos e janelas para gerar uma navegação pela aplicação simples e objetiva. Para isto foram usadas ferramentas da palete *Node-red-dashboard* que estão explicadas na Tabela 12.

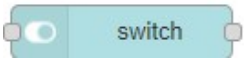
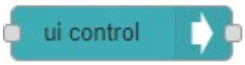
Node	Descrição
	Serve para acrescentar imagens ou endereços URL ao *dashboard*.
	Acrescenta um interruptor á interface, ou seja, foi usado para captar a informação de ligar e desligar a bomba de calor pelo utilizador.
	Funciona como uma entrada de dados numéricos, ou seja, foi usado para captar informação da temperatura desejada para a água do reservatório da bomba de calor.
	Serve para apresentar texto no *dashboard*, tal como os possíveis prolemas e causas de algum mau funcionamento ou anúncios de paragens de emergência devido a falhas mais graves.
	Mostra a mensagem numérica que recebe de forma criativa e apelativa na interface, especificando igualmente o valor numérico recebido.
	Cria um gráfico no *dashboard* para mostrar a variação dos valores dos sensores em função do tempo. É importante para saber em que ponto foi dado o alerta, e há quanto tempo a bomba de calor está em mau funcionamento.
	Gera avisos *pop-up* de mau funcionamento da bomba de calor, como, por exemplo, “Temperatura do compressor elevada” ou “Aumento brusco da temperatura do permutador”.
	Usado para navegar entre páginas da aplicação assim que lhe chega a correta instrução na *payload* da mensagem.

Tabela 12: Nodes utilizados na construção do Dashboard e respetiva descrição.

A interface com o utilizador está criada e é preciso disponibilizá-la na *Web* para que possa ser acedida em qualquer parte do Mundo com acesso à *Internet*. Isto foi conseguido de maneiras diferentes para cada controlador usado neste trabalho, uma vez que um suporta mais capacidades que o outro e, por isso, não necessita de tantas ferramentas suplementares para cumprir os objetivos. Em primeiro lugar são abordadas as técnicas para o microcontrolador e, de seguida, para o microcomputador, começando assim pelo maior número de recursos utilizados.

O microcontrolador ESP8266 é um dispositivo de baixo desempenho que não suporta o Node-Red para a criação da interface e, por isso, teve de se recorrer a um servidor online para realizar o código. Podia surgir um problema na disponibilização dos dados online, mas, como anteriormente foi referido, esses dados já estão num broker com acesso pela *Internet*, sendo possível usar o *node-red online*. A ferramenta utilizada foi o IBM Cloud, uma plataforma que oferece alguns serviços gratuitos para computação, armazenamento e de desenvolvimento de aplicações na *Web*, sem precisar de equipamentos externos. Primeiramente foi criada uma *app* Node-Red e configurados todos os parâmetros essenciais para a sua utilização. Depois, bastou iniciar a aplicação e todo o ambiente de trabalho Node-Red se assemelhava ao instalado num sistema operativo Windows ou Linux, fazendo com que a implementação do código previamente realizado fosse simples. Para interagir com a interface basta seguir o link "https://monitorizacaobombadecalor1.eu-gb.mybluemix.net/ui" num *browser* com *Internet*.

O microcomputador Raspberry pi 4 é um equipamento que utiliza Linux, de forma independente de qualquer outro computador, tornando possível a instalação do node-red no próprio sistema operativo através da linha de comandos. Após a instalação inicia-se o *software* com a linha "*node-red-start*" e, a partir de um qualquer *browser* do equipamento, pode-se ter acesso ao ambiente de trabalho habitual e fazer a HMI que se deseja. Inicialmente foi utilizado o URL "http://localhost:1880" para trabalhar a aplicação onde se construiu toda a interface com o utilizador, sendo que esta não poderia ser a solução final pois não estava disponível para toda a *internet*, mas apenas para os equipamentos conectados à rede local. A solução para esta limitação passaria por pôr disponível o *localhost* do Raspberry pi em qualquer dispositivo *online*. Para isso recorreu-se ao ngrok. Esta ferramenta cria um URL onde concede um caminho para que qualquer dispositivo possa utilizar um *localhost* de outro através da *Internet*, tornando-a muito útil para experimentar sites e aplicações. O *software* oferece vários pacotes, mas para este trabalho foi utilizado o gratuito que apenas fornece um acesso de cada vez e um URL aleatório, sendo o suficiente para esta fase de testes. Para aceder à HMI é necessário recorrer a um link do tipo "http://a02796d60bfc.ngrok.io/ui".

- **Mqtt Dashboard – IoT and Node-Red controller**

Apesar de já haver forma de interagir com a bomba de calor a partir de qualquer dispositivo fixo ou móvel que possua um *browser*, assim que o utilizador abandona o site não há indicações de um possível mau funcionamento. Apenas é enviado um email, mas que, muitas vezes, não é imediatamente visível. Assim sendo, foi usada uma aplicação da Playstore, a "Mqtt Dashboard – IoT and Node-Red controller", que permite criar um ambiente de trabalho próprio adicionando vários objetos, como botões e *inputs* e *outputs* de dados, tendo acesso diretamente ao *broker* para criar e modificar *topics*. Pode-se considerar uma forma intuitiva e fácil de trabalhar com recursos básicos,

mas suficientes para a primeira abordagem do problema em questão. Esta aplicação não é tão completa como a do Node-Red, mas pode estar constantemente a ser executada em segundo plano e enviar notificações para que haja um aviso imediato no *smartphone* ou *tablet* assim que se deteta uma falha.

Esta solução foi possível de implementar sem problemas nem no microcontrolador ESP8266 nem no microcomputador Raspberry pi, pois ambos já comunicavam com o Node-Red através do protocolo MQTT, e esta aplicação apenas funciona como mais um terminal que publica e subscreve dados no *broker*.

4.2.4. Algoritmo preditivo

O algoritmo preditivo implementado nesta dissertação teve por base uma aprendizagem supervisionada, como era de esperar, uma vez que o objetivo é detetar e prever falhas no sistema da bomba de calor. Como já referido, qualquer previsão precisa previamente dos valores apresentados no passado e, por isso, é necessário recorrer a uma base de dados para o armazenamento de toda a informação proveniente dos controladores, sendo que para uma maior assertividade na previsão é utilizado um maior número de dados.

Não foi possível localizar o armazenamento dos dados no microcontrolador ESP8266 pois este não está preparado para a instalação de programas para o efeito; então, recorreu-se novamente ao IBM Cloud para que este gerasse um local *online* para guardar os valores. A base de dados utilizada foi IBM Db2 on Cloud, que fornece gratuitamente 200MB de capacidade (suficiente para testes), com alto desempenho e linguagem SQL para que o armazenamento e a consulta dos dados esteja padronizada e seja de fácil acesso. Esta ferramenta garantiu que os valores dos sensores sejam guardados *online* e acessíveis por qualquer parte da *Internet* que possua as devidas credenciais para os requisitar.

Para o microcomputador Raspberry pi a base de dados foi criada no próprio *localhost*, pois sendo um microcomputador tem processamento e memória suficiente para que seja instalado um programa de gestão de dados. A ferramenta utilizada foi o MySQL, muito popular neste ramo da informática, sendo capaz de ajudar qualquer problema com a gestão de dados, garantindo um alto desempenho de forma económica. Organiza-se por bases de dados constituídas por tabelas com informação, sendo possível com uma linha de instrução extrair um valor específico ou milhares de valores. É bastante poderosa e foi usada para a gerir a grande quantidade de dados que foram recolhidos por este controlador, [18].

Nesta fase já se tem acesso aos valores antigos para que, a partir deles, possam ser previstos valores que ainda não foram registados. O algoritmo em si mesmo não é diferenciado pelos

controladores e, por isso, não é abordado em duplicado. A regressão linear, explicada de seguida, foi a técnica escolhida para prever dados futuros.

- **Regressão linear**

A regressão é baseada em dependências, e é usada para responder a perguntas como "o número de alunos influencia a classificação média da turma?", ou "qual é a próxima erupção de um vulcão baseando-se na duração da atual?". A regressão linear é usada em muitos problemas e pode ser considerada uma análise estatística. Por isso, a primeira etapa será resumir ao máximo os dados observados colocando-os em gráficos apropriados para que a visualização seja simples e útil, [19]. Os gráficos bidimensionais utilizados foram construídos com o tempo no eixo das abcissas e os valores dos sensores no eixo das ordenadas, como se observa na Figura 21.

Figura 21: Evolução temporal da temperatura do compressor.

A componente gráfica da variável foi obtida através do node-red, e é apresentada ao utilizador no histórico da aplicação para que possa haver uma análise convencional do comportamento dos sensores.

Com os pontos definidos no plano passou-se para a próxima etapa, que é a obtenção da reta que melhor aproxima todos os pontos obtidos. A equação desta reta tem a forma y=mx+b, onde m é o declive, b é a ordenada na origem e y e x são as ordenadas e as abcissas respetivamente. A reta no plano vai ter erros associados a cada ponto, mas que, se não forem muito significativos, não influenciam o propósito do trabalho. A expressão para chegar ao declive da reta a partir dos dados recolhidos é:

$$m = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

- x – Abcissas, hora do registo
- y – Ordenadas, valor do registo
- $\bar{x}$ – Média das horas do registo
- $\bar{y}$ – Media dos valores do registo
- n – Número total de registos

- m – Declive da reta

Obtendo o declive apenas fica em falta o ponto de intercessão com o eixo das ordenadas para ter a equação da reta mais representativa dos dados. Para isso, recorreu-se à equação, e assim completou-se a segunda fase da regressão linear.

$$b = \bar{y} - m\bar{x}$$

- $\bar{x}$ – Média das horas do registo
- $\bar{y}$ – Media dos valores do registo
- m – Declive da reta
- b – Intercessão da reta com o eixo das ordenadas

Encontrada a equação da reta da regressão linear estão criadas as condições para passar à etapa final do algoritmo preditivo, que consiste em prever os valores que irão aparecer nos sensores em alguns registos no futuro. Em primeiro lugar foi inserida a reta obtida no gráfico, dos pontos tal como ilustra a Figura 22 que foi perlongada no tempo, ou seja, depois do último registo no eixo das abcissas.

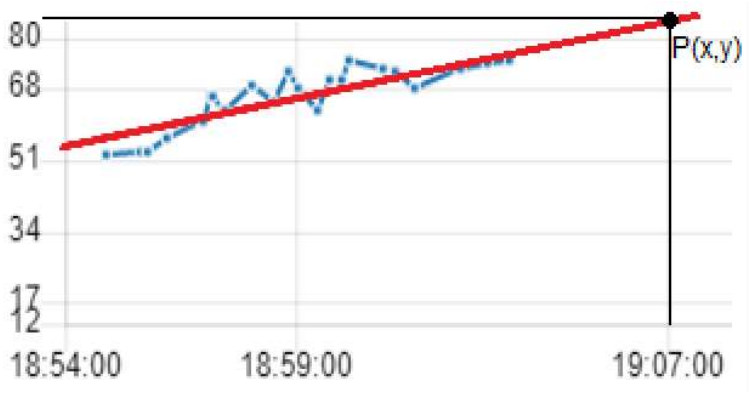


Figura 22: Evolução temporal da temperatura do compressor com a reta preditiva.

Na Figura 22 está assinalado um ponto P, coincidente com a reta da regressão, que representa a previsão do valor que se pretende obter. Neste caso, a aquisição dos registos foi feita até às 19:05:00 e foi previsto o valor que o sensor poderá adquirir às 19:07:00, estimando assim a temperatura do compressor passados dois minutos após o último registo. Quando a previsão de temperatura ultrapassa um limite especificado pelo programador é enviada uma notificação, ou um email, a avisar o utilizador que está prestes a ocorrer uma possível falha na bomba de calor.

Capítulo V

Resultados

Neste capítulo são apresentados os resultados do trabalho. Em primeiro lugar é feito um apanhado de todo o desenvolvimento recorrendo a fluxogramas para representar cada solução estudada e assim, esclarecer o fluxo de mensagens, o local das ferramentas utilizadas e em que etapa se insere cada componente que foi usado. Em segundo lugar, o algoritmo preditivo é novamente abordado para saber quais as previsões que são criadas e qual a sua utilidade, referindo variáveis preditivas e não preditivas deste trabalho. Por fim, as interfaces são explicadas, incluindo os menus implementados numa e noutra e as mensagens e notificações por elas criadas. No final, são indicadas as vantagens e desvantagens das duas soluções propostas, extraindo informações acerca da que se adequa a este sistema ou a outro similar.

5.1. Fluxogramas das soluções propostas

Um fluxograma é um diagrama de fácil compreensão de um processo ou algoritmo criado para um certo propósito. A sua utilização permite descomplicar e apresentar códigos que, de outra forma, tornar-se-ia aborrecido e confuso de compreender. Neste caso, o diagrama utilizou a Linguagem de Modelação Unificada, UML, para resumir e compactar as ações de implementação de ambas as soluções propostas, [20].

5.1.1. Solução para o microcontrolador ESP8266

O protótipo do microcontrolador anteriormente abordado faz referência a várias ferramentas extra ao equipamento para criar a interface, armazenar dados e fazer as previsões. Na Figura 23 são ilustrados esses recursos sequencialmente, transmitindo o local e a etapa onde foram usadas. Os fluxogramas foram criados recorrendo ao LucidChart, que é uma ferramenta *Web* que permite desenhar diagramas e gráficos, contribuindo para apresentações mais explícitas e precisas.

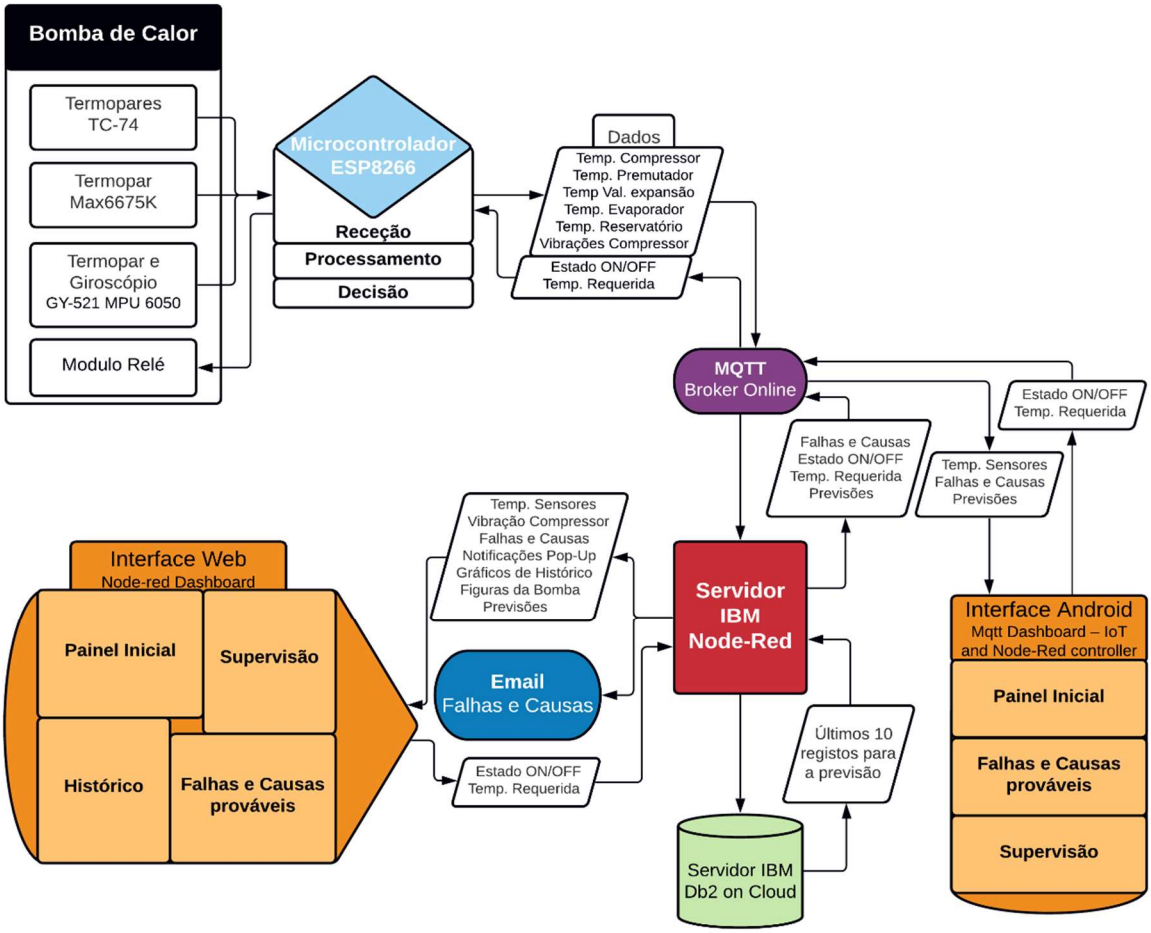


Figura 23: Fluxograma da solução para o microcontrolador ESP8266.

O diagrama começa no canto superior esquerdo com a bomba de calor a variar as temperaturas no seu circuito. Consequentemente, os sensores fazem a captação dessa variação e enviam-na para o microcontrolador. Após a receção e o processamento dos dados vem a decisão, que passa por saber se as temperaturas e as vibrações estão dentro dos limites estipulados para o bom funcionamento do equipamento. A informação é publicada num *broker online* e torna-se acessível em qualquer parte do Mundo, sendo daí que o IBM Cloud Node-Red retira os registos dos sensores. Os dados são processados no Node-Red, armazenados na IBM Db2 on Cloud, publicados na interface Node-Red Dashboard e usados para gerar mensagens de falhas e suas prováveis causas. Estas mensagens podem ser enviadas por email, ou publicadas na interface, ou simplesmente aparecerem numa notificação no ambiente interativo. Para além disto ainda são recebidos da base de dados os dez últimos valores registados para abastecer o algoritmo preditivo e, com isto, gerar novas notificações preditivas. A interface Android é fornecida com o estado dos sensores, falhas e prováveis causas dos problemas, e pelas previsões. Por fim, é observado o sentido oposto do fluxo da informação, sendo geradas nas interfaces mensagens com o estado *ON/OFF* da bomba de calor e

a temperatura desejada do reservatório. Esta informação é publicada no *broker online* pelo Node-Red e subscrita no Arduino instantaneamente, para que este possa fazer as operações necessárias para ligar ou desligar a bomba de calor, aumentando ou diminuindo a temperatura da água.

5.1.2. Solução para o microcomputador Raspberry pi 4

Nesta solução o armazenamento e o processamento dos dados, tanto em *Python* como no Node-Red, foram realizados no próprio microcomputador, pois este tem um poder informático muito superior ao microcontrolador ESP8266, que possibilita esta forma de trabalho. Na Figura 24 está representado o caminho percorrido pelas variáveis tanto fora como dentro do Raspberry pi.

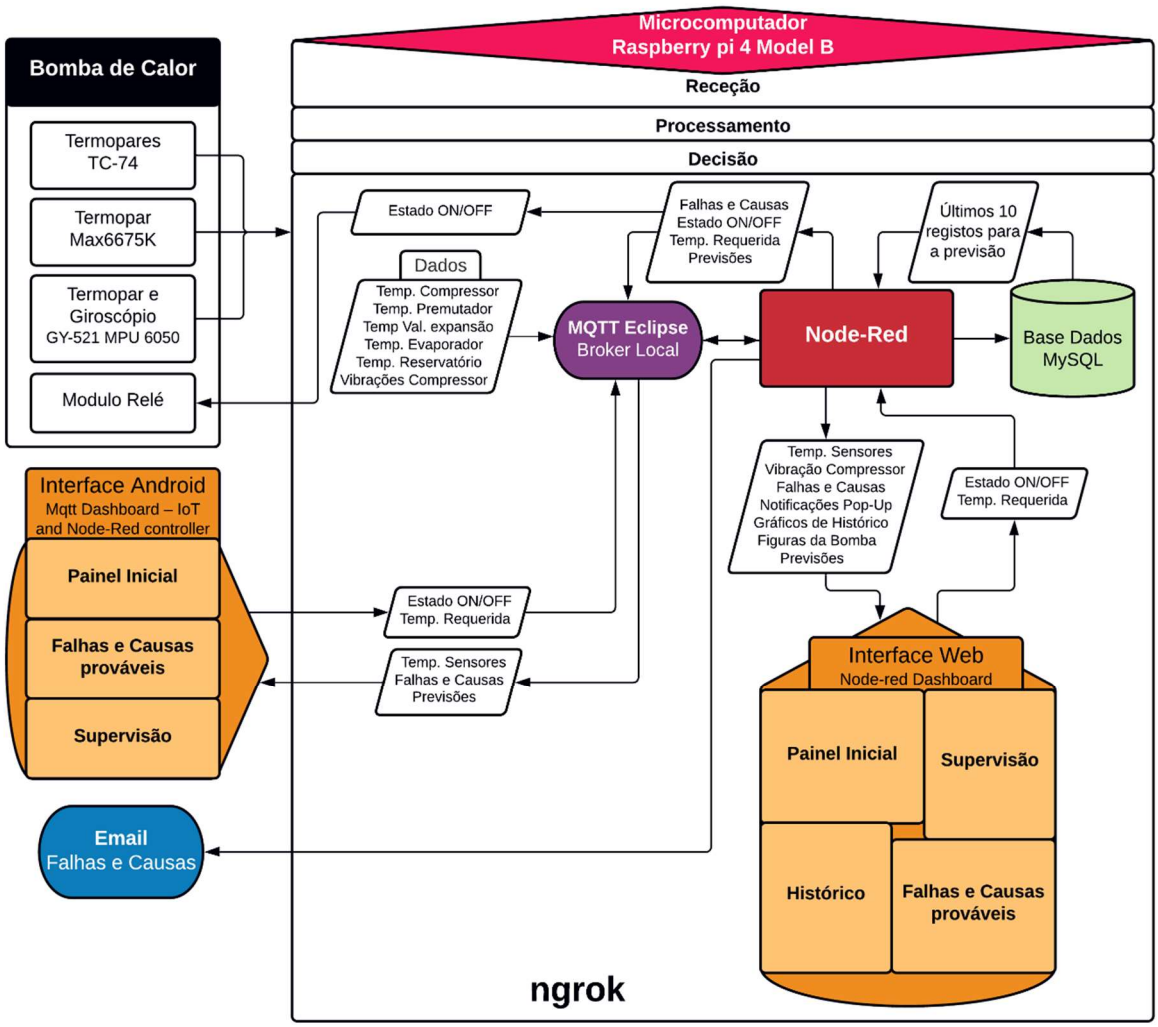


Figura 24: Fluxograma da solução para o microcomputador Raspberry pi 4.

À semelhança da solução anterior, o controlador recebe os dados dos sensores pelos protocolos de comunicação correspondentes e processa-os de forma a obter as temperaturas e vibrações do sistema. Através de um programa em *Python*, decide se os parâmetros estão dentro dos limites para o bom funcionamento do sistema e envia-os para um servidor *broker* previamente instalado no microcomputador. Sendo este *broker* local, ou seja, apenas visível na rede *WiFi* a que está conectado, teve de ser acionado o ngrok, tornando o *localhost online* e acessível através de um URL gerado pelo *software*. A partir deste momento tanto o servidor MQTT como o Node-Red e a base de dados MySQL estão acessíveis por qualquer dispositivo com recurso à *Internet*. O Node-Red recebe os dados a partir do *broker*, armazena-os no MySQL e processa-os para gerar mensagens de falhas e causas prováveis de problemas na bomba de calor. As variáveis dos sensores e estas mensagens são enviadas por email e para a interface *Web* para serem apresentadas ao utilizador, podendo assumir um formato de notificação *pop-up*. Para além disto, as falhas e causas prováveis são publicadas no *broker* juntamente com as previsões geradas através dos últimos dez valores recebidos da base de dados. Isto serve para que a interface *Android* tenha acesso a esta informação e a disponibilize ao utilizador. No entanto, ao contrário da primeira solução, as mensagens precisam de correr no sentido oposto, mas já não é essencial que passem pelo broker novamente, pois a partir do Node-Red é possível controlar os GPIO. Assim sendo, a interface *Web* transmite o estado ON/OFF da bomba de calor e da temperatura desejada para a água do reservatório ao Node-Red e este executa imediatamente as operações de acionamento do devido relé, não deixando de ser importante a publicação no broker destas alterações para haver sincronização com a interface *Android*, que comunica apenas pelo *broker*.

5.2. Previsões do algoritmo

As previsões feitas no âmbito deste trabalho assumiram um papel mais informativo e de alerta que para gerar ações concretas no funcionamento da bomba de calor, uma vez que são constantemente atualizadas e podem não corresponder à total realidade. A sua relatividade pode torná-las enganadoras e, consequentemente, proceder a ações que não são as mais apropriadas à situação. Porém, não deixa de ser uma mais valia receber a informação de que algo no sistema pode estar a caminhar para um funcionamento deficiente, sendo este o motivo para a adoção das previsões neste trabalho. De qualquer modo, exploradas as vias para a obtenção, armazenamento e tratamento da informação, o sistema de previsão, emissão de alertas e desencadear de ações para cada sistema concreto pode ser amplamente melhorado na base de um quadro de deteção de possíveis maus funcionamentos, das causas mais prováveis para a sua ocorrência e dos sensores mais adequados para a sua deteção.

Os valores dos sensores que permitiram a implementação deste algoritmo preditivo foram as temperaturas no compressor, condensador, válvula de expansão e evaporador, pois costumam apresentar registos essencialmente constantes, com variações mínimas. As vibrações medidas não foram utilizadas porque o seu valor tem variações muito acentuadas e, ao fazer a regressão linear esta, possivelmente, não seria representativa, pondo em causa as previsões feitas pelo algoritmo e dando alertas sem motivo. Tal significa que há todo um trabalho de análise das vibrações do compressor que carece de ser feito, de modo a estabelecer um padrão que possa ser útil para o objetivo de prever mal funcionamento do compressor a partir da sua medição em tempo real. Posto isto, foram definidos os intervalos limite de temperaturas em que as previsões de cada ponto no circuito. Assim que se registasse um valor fora desses intervalos seria gerada uma mensagem para ser enviada por email e apresentada em ambas as interfaces. Na Tabela 13 são apresentados os intervalos de temperatura utilizados no algoritmo.

Compressor (°C)	Condensador (°C)	Válvula de expansão (°C)	Evaporador (°C)
[0; 100]	[10; 50]	[-5; 25]	[10; 30]

Tabela 13: Intervalos limite de temperaturas usados no algoritmo preditivo.

5.3. Interface *Web*

A interface *Web* foi construída, como já referido, a partir do Node-Red Dashboard, que permite fazer um menu dividido em várias páginas, estando organizadas em blocos de informação para que seja proporcionada uma experiência de utilização intuitiva e dinâmica. O menu foi dividido em quatro submenus com diferentes objetivos, e por conseguinte, diferentes objetos que servem para informar estados dos sensores ou para interagir com a bomba de calor. De seguida são abordadas as várias páginas, esclarecendo os seus propósitos e cada objeto que foi incluído. No final são inumeradas as mensagens *pop-up* possíveis de aparecer e o formato do email que será enviado para a conta pré-definida.

5.3.1. *Login*

O início da aplicação tem lugar com a página do login, onde é pedido um utilizador e uma palavra chave para que seja concedido acesso à interface. As credenciais corretas foram fornecidas ao programa antecipadamente, e é num *node* função que são comparadas com os i*nputs*. Após o teste, uma nova página é mostrada já com objetos para a interação. A Figura 25 mostra a disposição da etapa do *login*.

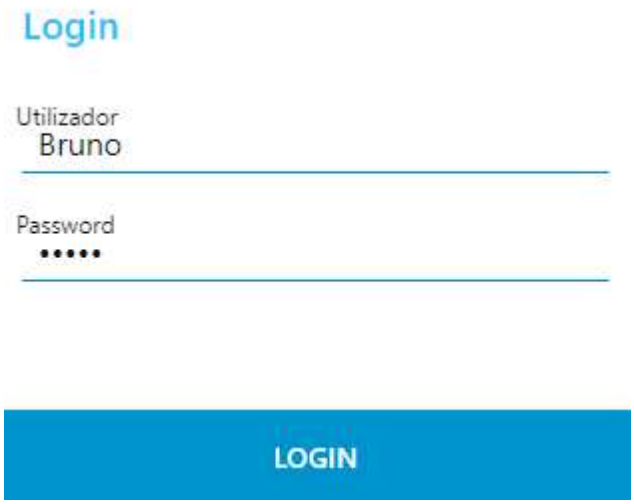


Figura 25: Página de Login da interface Web.

Quando uma palavra chave ou utilizador é inserido incorretamente, abaixo do objeto *"Password",* é escrita a mensagem "Utilizador ou Password incorreta" e o operador pode tentar aceder novamente as vezes que entender. Uma vez que a interface apenas serve para monitorizar uma bomba de calor não houve interesse em explorar as questões de segurança com mais detalhe e fica assim concluída a etapa de acesso à interface *Web*.

5.3.2. Monitorização

Depois de concluir o *login*, o utilizador é direcionado para uma nova página com o nome Monitorização. Aqui dá-se o início da aplicação, e é onde há mais interação com a bomba de calor pois é possível ligar e desligar, informar qual a temperatura desejada para a água do reservatório e visualizar qual a temperatura atual da água. Para além disso, através de um painel de navegação, fica garantida a ligação com as restantes páginas. Na Figura 26 apresenta-se o *layout* adotado nesta parte da interface.

Figura 26: Página de Monitorização da interface Web.

Esta parte do menu está organizada em três blocos. O primeiro é constituído por um botão *ON/OFF*, para ligar e desligar a bomba de calor através do modulo relé, um input de temperatura, para indicar a temperatura desejada para a água do reservatório na bomba de calor, e um indicador da verdadeira temperatura da água no reservatório (fornecido pelo sensor MAX6675 K). Quando a paragem de emergência é acionada também é possível visualizar um alerta indicativo por baixo do objeto *ON/OFF*. Esta paragem de emergência apenas é acionada quando os valores das temperaturas ultrapassam os limites expostos na Tabela 13. O segundo bloco foi apenas ocupado por uma imagem ilustrativa da bomba de calor a monitorizar. Por fim, à esquerda encontra-se o painel de navegação com a indicação da página atual e com mais três botões, um para cada página dos menus que são abordados a seguir.

5.3.3. Supervisão

Um dos submenus está relacionado com a supervisão, que tem os valores dos sensores localizados no sistema da bomba de calor em tempo real. Esta página é útil para saber qual o estado da bomba de calor durante o seu funcionamento, para que sejam visíveis as consequências da manutenção no instante em que as ações são executadas. Apenas tem objetos informativos que não podem ser alterados pelo utilizador. Na Figura 27 está ilustrada a interface da supervisão do sistema.

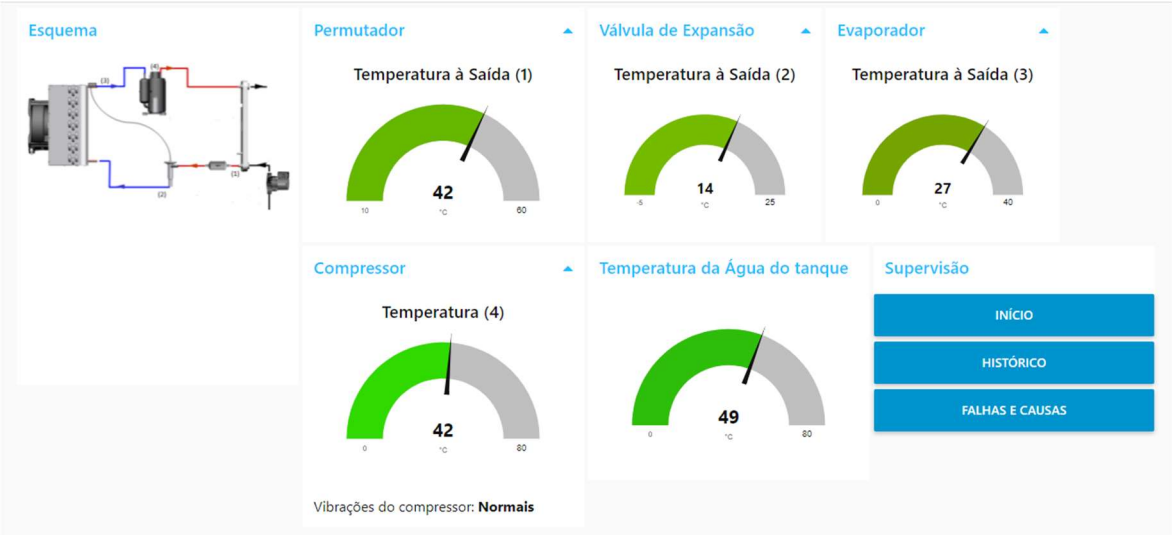


Figura 27: Página de Supervisão da interface Web.

Como mostra a Figura 27, esta página está dividida em sete blocos: um para cada sensor, um dedicado à navegação da aplicação e outro que mostra o circuito do sistema da bomba de calor com todos os seus componentes, e a localização de cada sensor. No canto superior esquerdo da Figura 25 é observável este último bloco, e à direita cada bloco com a informação de cada sensor. TC-74 do condensador, da válvula de expansão e do evaporador. Em baixo, mostra a temperatura juntamente com as vibrações, normais ou irregulares, do compressor registadas pelo sensor GY-521 MPU 6050, seguido da temperatura da água do reservatório e do painel de navegação.

5.3.4. Histórico

O Histórico é outra página informativa que fornece ao utilizador os valores registados nas últimas duas horas de utilização da bomba de calor, para que se possa saber se a falha ocorreu instantaneamente ou se se tem vindo a prolongar no tempo. Isto pode poupar trabalhos de manutenção e permitir identificar possíveis falhas nos sensores, pois a temperatura não varia de forma abrupta e, se tal acontecer, os sensores podem estar danificados, sendo necessário proceder à sua substituição. A Figura 28 mostra a configuração gráfica da página do Histórico.

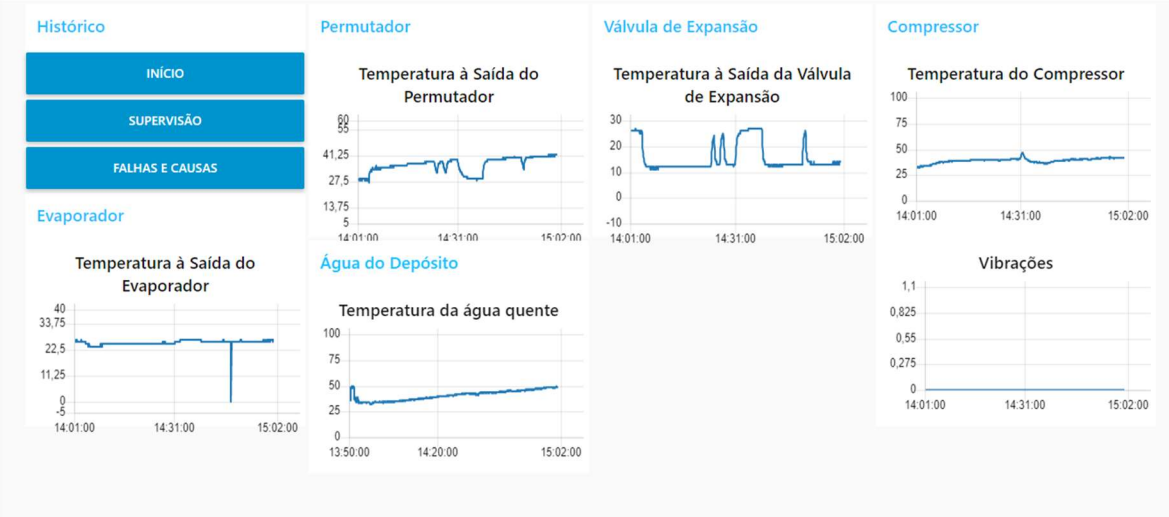


Figura 28: Página de Histórico da interface Web.

No canto superior esquerdo encontra-se o painel de navegação, com botões que levam o utilizador para as restantes páginas. À direita e por baixo deste encontram-se cinco blocos com as evoluções temporais dos registos. O primeiro grupo tem os valores da temperatura à saída do condensador e o segundo da válvula de expansão. O terceiro bloco apresenta dois gráficos, a temperatura do compressor e as vibrações, que apenas variam entre 0 e 1, sendo que o 0 significa que as vibrações estão normais e 1 que estão irregulares. Na segunda linha são expostos mais dois blocos, um com o gráfico da temperatura à saída do evaporador e o outro com a temperatura da água do reservatório.

5.3.5. Falhas e prováveis causas

A página das falhas e prováveis causas tem novamente apenas um papel informativo, e não possui nenhuma interação com a bomba de calor, fazendo com que os objetos utilizados sejam do tipo texto. Este submenu é importante, pois é nele que são diagnosticados os problemas detetados através dos valores medidos pelos sensores, prevendo qual é a falha do sistema e sugerindo a sua causa, para que assim o técnico de manutenção se desloque ao local preparado com o equipamento necessário para a operação de manutenção preventiva. Assume assim um papel mais técnico, não tendo grande relevância para o utilizador. Na Figura 29 é possível observar esta página da aplicação.

Figura 29: Página de Falhas e prováveis causas da interface Web.

Como mostra a Figura 29, a página está dividida em seis grupos de objetos. Os quatro objetos centrais remetem para o estado de cada componente do circuito, à esquerda para o painel de navegação similar ao das páginas anteriores e à direita para o histórico de falhas. Os blocos centrais apenas podem mostrar duas mensagens, uma quando o componente não apresenta nenhuma anomalia, por exemplo "Compressor operacional", e quando o seu funcionamento é deficiente, por exemplo "Falha na válvula de expansão, visualizar causas em histórico". O histórico tem preparadas várias mensagens que, para além de informar que falhas ocorreram também indicam as possíveis causas. Estas mensagens encontram-se na Tabela 14.

Componente Avariado	Mensagem
Condensador	Temperatura alta: "Bomba de água danificada, sistema de circulação de água entupido, ou condensador danificado" Temperatura baixa: "Temperatura baixa, aquecimento em curso."
Válvula de expansão	Temperatura alta: "Falta de fluido refrigerante, colmatação ou falta de ventilação do evaporador" Temperatura baixa: "Temperatura baixa."
Evaporador	Temperatura alta: "Temperatura alta." Temperatura baixa: "Colmatação ou falta de ventilação"
Compressor	Temperatura alta: “Falta de fluido refrigerante, ou falta de óleo lubrificante." Temperatura baixa: "Temperatura baixa, aquecimento em curso."

Tabela 14: Mensagens do histórico, falhas e prováveis causas.

Na Tabela 14 são apresentadas as mensagens a que o utilizador vai ter acesso quando um sensor envia temperaturas demasiado altas ou baixas. Esta decisão é tomada no controlador, e o node função do Node-Red apenas decide se publica uma ou outra mensagem no histórico. Esta opção de dividir o processamento da informação prende-se no facto de reduzir o número de publicações no *broker* e, assim, poupar mensagens quando estas são iguais, ou seja, quando a temperatura está demasiado alta é indiferente que esteja 10°C ou 20°C a mais do que é permitido e, por isso, apenas é enviada uma vez a informação de que ultrapassou o limite esperado. Se a temperatura voltar aos valores normais voltam a ser enviadas mensagem regulares.

5.3.6. Notificações

As notificações de mau funcionamento foram implementadas recorrendo ao *node notification* que permite criar um aviso *pop-up* nos limites verticais ou horizontais da aplicação. Estes avisos são uma parte essencial do sistema de monitorização para que o utilizador seja informado das avarias e possa de imediato averiguar as falhas e as suas prováveis causas. Uma vez que apenas aparecem na interface *Web*, á qual nem sempre o utilizador está atento, foi necessário encontrar uma solução que o alertasse de funcionamentos indevidos, mesmo quando não estava a usar a aplicação. Para isso, foi utilizado o *node email*, para enviar um email a notificar da existência da falha e, adicionalmente a identificada provável causa para um diagnóstico breve e eficaz. Este email é composto pelas mesmas mensagens da Tabela 14. Na Tabela 15 estão identificadas as mensagens *pop-up* que podem aparecer ao utilizador durante a utilização da aplicação.

Componente Avariado	Notificações
Condensador	Temperatura alta: “Mau funcionamento do Condensador” Temperatura baixa: “Condensador frio! Aguarde...” Previsão: “Brusco aumento de temperatura do Condensador”
Válvula de expansão	Temperatura alta: " Mau funcionamento da Válvula de Expansão" Temperatura baixa: " Bomba de Calor fria! Aguarde...” Previsão: “Brusco aumento de temperatura da Válvula de Expansão”
Evaporador	Temperatura alta: “Evaporador quente! Aguarde...” Temperatura baixa: “Mau funcionamento do Evaporador" Previsão: “Brusca diminuição de temperatura do Evaporador”
Compressor	Temperatura ou vibrações altas: “Mau funcionamento do Compressor” Temperatura baixa: “Bomba de Calor fria! Aguarde...” Previsão: “Brusco aumento de temperatura do Compressor”

Tabela 15: Notificações pop-up da interface Web.

As notificações são apenas de aviso e, por isso, para saber as suas causas é necessário recorrer à página das Falhas e Causas no bloco do Histórico, onde se explicam as falhas e as suas possíveis causas.

No assunto do email enviado consta “Aviso de mau funcionamento da Bomba de Calor” e no corpo da mensagem uma derivação do tipo “Possíveis causas do mau funcionamento: Falta de fluido refrigerante, colmatação, ou falta de ventilação do evaporador” com a falha verificada no momento, que, neste caso, seria uma falha na válvula de expansão.

5.4. Interface *Android*

Apesar de a interface *Web* estar acessível a partir de todos os dispositivos com acesso à *Internet* as notificações podiam continuar a ser um problema uma vez que o utilizador nem sempre está atento à interface nem ao email. Por isso houve a necessidade de criar uma aplicação *Android* que, ao correr em segundo plano num qualquer *smartphone* ou *tablet*, envia notificações para o aparelho a alertar para as falhas e a chamar a atenção do utilizador. Para além disso o utilizador teria um acesso mais cómodo à bomba de calor sem necessitar de usar um URL. Como foi referido anteriormente, para gerar a interface *Android* recorreu-se a uma *app* da *PlayStore,* a "Mqtt Dashboard – IoT and Node-Red controller", que acede ao *broker* diretamente para retirar as informações necessárias. O *login* nesta interface não foi possível de implementar, mas como o acesso ao *broker* é restrito fica assegurada a privacidade da aplicação. Na Figura 30 é possível observar a organização da *app*.

Figura 30: Interface Android

Nesta aplicação foi fornecido o *broker* a que se conectou e os *topic* que subscreveu, organizando a sua apresentação em 4 grupos. O primeiro são os atalhos, os objetos mais importantes são mostrados no topo, ou seja, um botão para ligar e outro para desligar a bomba de calor, um *input* para a temperatura desejada e, por fim, um objeto de texto para mostrar os alarmes acionados. Depois são apresentados novamente no bloco da "monitorização da bomba de calor" os botões de ligar e desligar e o *input* da temperatura desejada para a água do reservatório. Segue-se o bloco da "Supervisão", com os valores dos sensores do sistema da bomba de calor em tempo real e o bloco das "Falhas e Causas", com os alarmes que foram acionados e as respetivas causas.

5.5. Comparação final

Neste subcapítulo são comparadas as duas soluções propostas nesta dissertação, referindo as vantagens e as desvantagens de ambas pois, apesar de as interfaces finais e os resultados serem semelhantes, toda a implementação e custos associados divergem, tornando as soluções mais ou menos apelativas. Podem-se então comparar as principais caraterísticas de cada solução para o problema proposto nesta dissertação na Tabela 16.

	Microcontrolador ESP8266	Microcomputador Raspberry pi 4
Implementação	Fácil adaptação ao *software* de trabalho; instalado num computador externo.	Ambiente de trabalho Linux, com todas as funcionalidades de um computador.
	Programação num computador externo.	Programação feita no controlador, ou num outro computador através do VNC.
	Programação em linguagem C#.	Programação em linguagem *Python*, com a possibilidade de instalar compiladores C#.
	Suporte técnico superior e ao encontro das necessidades (Tutoriais).	Maior dificuldade no acesso ao suporte técnico desejado (Tutoriais).
	Bibliotecas de qualidade e precisas.	Elevado número de bibliotecas de simples manuseamento.
	Menor número de pinos GPIO's, sendo necessária a sua melhor gestão.	Bastantes pinos GPIO para ligar os barramentos.
	Incompatibilidade com a instalação do Node-Red; necessário recorrer a externos.	Possibilidade de instalar programas para o projeto (Node-Red, ngrok, MySQL).
	Impossibilidade de usar uma base de dados própria sem recorrer a externos.	Base de dados guarda a informação no próprio equipamento.
	Capacidades de processamento suficientes para o projeto.	Capacidades de processamento excessivos atendendo aos requisitos do projeto.

Uso	Baixos consumos de energia. (< 0,5 W)	Consumos médios de energia. (3,5 W)
	Impossibilidade de acoplar um ecrã no projeto.	Compatibilidade com ecrã hdmi para uma interface na bomba de calor.
	Apenas entrada Micro USB para o envio do código sem acesso à interface.	Entradas USB e de *Ethernet* para uma comunicação direta com a interface.
	Sem armazenamento de dados *offline*.	Armazenamento de dados *offline.*
	Acesso à bomba de calor *offline* impossível.	Acesso à Bomba de calor *offline*, recorrendo a um ecrã ou computador.
	Menor dimensão.	Maior dimensão.
	Mais lento a executar as operações	Operações executadas instantaneamente
	Uso das várias ferramentas externas utilizadas (Broker, Servidor IBM).	Uso de apenas de uma ferramenta externa (ngrok).
Custo	Equipamento mais barato. (≈5 €)	Equipamento mais caro. (≈60 €)
	Solução mais económica (≈20 €) *	Solução menos económica (≈80 €) *

Tabela 16: Análise comparativa das duas soluções propostas.

* Nota: Em acréscimo a estes custos existem as subscrições mensais das ferramentas online com vários planos que podem melhorar as soluções. Estas oferecem diversas vantagens como mais utilizadores com acesso às interfaces ou mais atualizações por minuto dos estados das variáveis, divergindo assim os seus custos.

Capítulo VI

Conclusões

A monitorização é uma área da engenharia que tem vindo a crescer nas últimas décadas, tornado o quotidiano do ser humano mais eficiente e cómodo melhorando o funcionamento dos equipamentos e prolongando a sua vida útil. A Sociedade cada vez mais ambiciona ter o controlo dos equipamentos que a rodeiam à simples distância de um clique e, por isso, é necessário abordar cada dispositivo individualmente para tornar esse sonho realidade. O trabalho apresentado neste documento foi a concretização deste desejo para as bombas de calor, uma vez que a sua presença no mercado tende a aumentar. Ainda assim, os desenvolvimentos feitos podem ser estendidos a praticamente qualquer sistema, recorrendo aos recursos mais adequados a cada caso.

Inicialmente, o estudo da bomba de calor a monitorizar foi essencial, identificando os parâmetros pertinentes para a deteção das falhas, e a forma de monitorização que melhor correspondia aos requisitos do sistema de monitorização a desenvolver. A maior dificuldade passou por saber quais as possíveis falhas que um sistema deste tipo pode apresentar e, posteriormente, provoca-las, verificando as consequências que induziam nos registos efetuados.

Após a escolha dos equipamentos utilizados, como sensores e interruptores, surgiu um novo dilema. O controlador do sistema podia ser tanto um microcontrolador como um microcomputador e, então, foi decidido implementar as duas soluções. No final deste processo, a comparação dos aspetos positivos e negativos de ambas revelou que os requisitos do sistema de monitorização a desenvolver não eram suficientes para decidir qual das implementações melhor satisfazia o problema proposto, uma vez que tanto uma como outra tinham pontos a favor e contra. Apesar disso é de notar as claras vantagens que a solução com o microcomputador Raspberry pi apresenta e que no final da análise podemos declarar que seria a mais indicada para cumprir com os objetivos propostos, ficando, contudo, muito mais dispendiosa.

Posto isto, esta dissertação teve como principal foco a monitorização de um qualquer equipamento comercial, doméstico ou industrial, tomado o exemplo de uma bomba de calor para abordar um problema prático, mas sempre com a intenção de poder ser simplesmente transposto para outros sistemas. As ferramentas utilizadas são acessíveis por qualquer um, sendo este trabalho uma espécie de guia para a implementação da *Internet of Things* em um qualquer meio. As duas formas de resolver o problema abrem os horizontes para qualquer projeto que tencione ser controlado pela *Internet*. A implementação num outro equipamento teria de se guiar por algumas etapas. A primeira etapa é essencial para o sistema a desenvolver pois é preciso estudar o equipamento a monitorizar e definir os parâmetros que melhor informam o seu estado. A segunda etapa tem o objetivo de encontrar os sensores que permitem a obtenção dos parâmetros a retirar e qual o controlador que melhor satisfaz os requisitos do sistema a monitorizar. Encontrando o *hardware* a utilizar, na terceira etapa, escolhe-se o *software* que garante a comunicação M2M e gera a interface com o utilizador. Por último, a quarta etapa consiste em analisar os dados provenientes do equipamento e processá-los

de maneira a gerar um algoritmo de prevenção de falhas e as suas prováveis causas. Seguindo este guia a monitorização em tempo real é alcançada para qualquer equipamento eletrónico que careça desta propriedade.

Na realização prática das soluções surgiram alguns problemas inesperados, que podem ter posto em causa os prazos de entrega. Uma situação que levou ao atraso foi a falta de revestimentos e conectores no protótipo inicial, que levaram a curto-circuitos e maus contactos, impossibilitando a correta transferência de dados entre sensores e controladores. Outro problema surgiu com a falta de um manual técnico da bomba de calor, que dificultou o acesso à HMI e, consequentemente, o funcionamento do aparelho. Para além disso, como o equipamento já não trabalhava há alguns anos e não teve as devidas manutenções, a bomba circulação acusou sinais de encravamento não cumprindo a sua função e requerendo tempo para reparações.

Relativamente a trabalhos futuros, propõe-se o desenvolvimento de uma aplicação *Android* para substituir a ferramenta fornecida pela *Playstore*, tornando o trabalho mais personalizado e profissional. Esta aplicação poderia utilizar melhor os recursos do *smartphone* e, assim, contribuir para uma monitorização efetiva. Outros trabalhos poderiam contemplar a implementação destas ferramentas noutros equipamentos e controlá-los a todos por meio de uma só interface, podendo ser implementado numa habitação para monitorizar os eletrodomésticos ou numa indústria para obter a produção de cada máquina das instalações. Para além destas propostas ainda seria interessante controlar cada componente da bomba de calor em vez de apenas ligar ou desligar esta.

As aplicações para a IoT são imensas, e cabe a cada criativo juntar as ferramentas disponíveis no mercado e implementar a sua solução para a resolução de qualquer problema de monitorização em tempo real, seja apenas com objetivo descritivo (*dashboard*), seja para emissão de alertas e interação com os próprios equipamentos.

Bibliografia

[1] D. J. MacKay, *Sustainable Energy — Without the hot air*, 2009.

[2] L. Gomes, “Datalogger Remoto - Monitorização e controlo de uma Bomba de Calor,” Universidade de Aveiro, 2013.

[3] Arduino, “What is Arduino?,” 2018. https://www.arduino.cc/en/Guide/Introduction (accessed Mar. 25, 2021).

[4] RASPBERRY PI FOUNDATION, “Raspberry Pi 4.” https://www.raspberrypi.org/products/raspberry-pi-4-model-b/ (accessed Mar. 25, 2021).

[5] NXP Semiconductors N.V., “I2C-bus specification and user manual,” 2014. doi: 10.1007/s10792-016-0274-8.

[6] Maxim Integrated, “MAX6675 Datasheet,” vol. 19, p. 8, 2014, [Online]. Available: www.maximintegrated.com.

[7] C. M. Kozierok, *The TCP/IP Guide : a comprehensive, illustrated internet protocols reference*. 2005.

[8] A. D. John D. Kelleher, Brian Mac Namee, *FUNDAMENTALS OF MACHINE LEARNING FOR PREDICTIVE DATA ANALYTICS*. Cambridge, Massachusetts London, England, 2015.

[9] A. M. M. dos Santos, “Algoritmos para manutenção preditiva, na industria 4.0,” 2018.

[10] M. Oliveira, “Aula 7 de Termodinâmica Aplicada, A segunda lei da termodinâmica e entropia,” pp. 17–24, 2019.

[11] Bosch Termotecnologia S.A, “Bosch Compress 3000 DW FO Specification _ Manualzz.pdf.” Aveiro, pp. 23–31, 2012.

[12] Mitsubishi Electric, “Refrigerant Compressors.”

[13] InvenSense Inc., “MPU-6000 and MPU-6050 Product Specification Revision 3.4,” p. 7, 2013, doi: 10.1093/fs/XXXIII.1.1.

[14] InvenSense Inc., “MPU-6000 and MPU-6050 Register Map and Descriptions Revision 4.2,” vol. 1, no. 408, p. 48, 2013, [Online]. Available: www.invensense.com.

[15] Microchip Technology Inc., “Tiny Serial Digital Thermal Sensor,” vol. 3, pp. 1–8, 2002.

[16] L. A. Carmen Gutierrez, Ignácio Fernández, Ana Viñuela, *Encicolpédia Basica do Estudante*. 2005.

[17] Arduino, “Wire Library,” 2019. https://www.arduino.cc/en/Reference/Wire (accessed Jan. 31, 2021).

[18] Oracle Corporation, “MySQL Products,” 2021. https://www.mysql.com/products/ (accessed Feb. 01, 2021).

[19] S. Weisberg, *Applied Linear Regression*, 3ª Edição. Hoboken, NJ, 2006.

[20] Lucid Software Inc., “O que é um fluxograma?,” 2021. https://www.lucidchart.com/pages/pt/o-que-e-um-fluxograma (accessed Jun. 25, 2021).