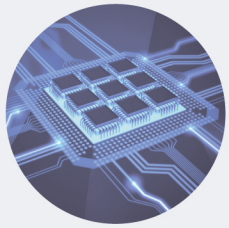
INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA DE SISTEMAS COMPUTACIONAIS CRÍTICOS



Controlo e Monitorização do Ar Interno

JORGE MANUEL E SILVA
outubro de 2023

P.PORTO

ISEP INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO

IACAM
Indoor Air Controlling and Monitoring

Jorge Manuel e Silva

Dissertação para obtenção do Grau de Mestre em Engenharia de Sistemas Computacionais Críticos

Orientador: António Barros

Júri
Presidente:
Luis Lino Ferreira, ISEP

Vogais:
Paulo Baltarejo Sousa, ISEP
António Barros, ISEP

Porto, 7 de outubro de 2023

Declaração de Integridade

Declaro ter conduzido este trabalho académico com integridade.

O trabalho apresentado neste documento é original e de minha autoria, e foi realizado no âmbito do Mestrado em Engenharia de Sistemas Computacionais Críticos.

Não plagiei ou apliquei qualquer forma de uso indevido de informações ou falsificação de resultados ao longo do processo que levou à sua elaboração, todas as referências foram reconhecidas e citadas na integra, e todo o texto foi originalmente produzido por mim (exceto quando devidamente anotado).

Declaro ainda que tenho pleno conhecimento do Código de Boas Práticas e Conduta do Instituto Politécnico do Porto.

Porto, Porto, 7 de outubro de 2023

(Assinatura conforme documento de identificação, ou, preferencialmente, assinatura digital)

Dedicatória

Uma das premissas da vida que levo é que a mesma é feita de momentos, sejam eles momentos positivos ou menos positivos. Contudo, cabe a cada um de nós fazer esses momentos acontecer porque sem eles mais tarde será impossível recordar esses momentos, essas vivências, esta vida.

Portanto, coleciono mais um momento, desta vez positivo, na minha vivência e no meu percurso académico que é a realização da minha dissertação.

Este momento proporcionou episódios deveras stressantes e outros mais satisfatórios e felizes, mas nessas circunstâncias estiveram por detrás pessoas incríveis e capazes de dar o devido apoio. Logo, essas pessoas também ficam marcadas neste momento.

É inesquecível o quão importante foi todo o teu suporte nesta minha etapa pois, em todos os instantes foste Tu que me apresentantes soluções e conselhos fundamentais para não perder o rumo. Desde o inicio que te foste tornando um alicerce e um porto de abrigo em tudo para me elucidar. Desta maneira, dedico uma parte deste nosso momento a Ti por todo o apoio, força e paz que me foste transmitindo.

Também não me posso esquecer de Ti pois, foi graças à tua vontade, persistência, orgulho e confiança em mim que este momento foi possível. Sempre acreditaste mais do que eu no meu potencial e isso foi crucial para chegar onde estou e vou chegar. Apesar da distância nunca me esqueço da tua força e agradeço por isso. Sempre me ensinaste a ser humilde e simples portanto, dedico uma parte a Ti também deste nosso momento.

Para mim sempre serás aquele menino pequenino e reguila apesar disso, também estiveste presente neste processo à tua maneira. Então, não posso nem me vou esquecer de Ti e assim também te dedico uma parte deste nosso momento.

Por fim, também não me posso esquecer de Ti que de uma forma ou outra deste o teu apoio. Da mesma forma, também dedico uma parte deste nosso momento a Ti.

Resumo

Manter a operação fiável e segura dos servidores de uma organização é uma tarefa árdua que exige uma contínua monitorização e controlo de diversos fatores. O controlo do ambiente de operação de um servidor (e.g. temperatura e humidade) é muito importante, pois o desvio das condições nominais de operação dos equipamentos pode comprometer os processos que deles dependem, com consequente impacto negativo para a organização.

Idealizar e prototipar um produto final é um processo trabalhoso, dado que as suas condições variam de organização para organização. Para cada caso é fundamental pensar, idealizar, planear e, por fim, executar.

Este projeto tem como principal objetivo monitorizar e controlar as condições ambientais presentes num bastidor através de sensores, atuadores. Os dados amostrados são integrados numa aplicação que determina as ações que permitem manter as melhores condições de operação e prevenir/detetar a ocorrência de acidentes de forma a minimizar a possibilidade de perda de recursos e dados críticos.

Palavras-chave: Centros de dados, Salas de servidores, Prevenção de incêndios, Monitorização e controlo do ambiente interior, Monitorização de servidores

Abstract

Maintaining reliable and secure operation of an organization's servers is an arduous task that requires continuous monitoring and control of several factors. Controlling the operating environment of a server (e.g. temperature and humidity) is critical, as deviations from the equipment's nominal operating conditions can compromise the processes that depend on them, with a consequent negative impact on the organization.

Idealizing and prototyping a final product is a laborious process, as its conditions vary from organization to organization. For each case, it is essential to think, idealize, plan and, finally, execute.

This project's main objective is to monitor and control the environmental conditions present in a rack through sensors and actuators. The sampled data is integrated into an application that determines the actions that allow maintaining the best operating conditions and preventing/detecting the occurrence of accidents in order to minimize the possibility of loss of resources and critical data.

Agradecimentos

Antes de tudo, um dos meus agradecimentos vai para o Instituto Superior de Engenharia do Porto (ISEP) por, durante 2 (dois) anos, me proporcionar um mestrado muito intuitivo e enriquecedor, agradecendo também a todo o grupo de docentes que fizeram acontecer o Mestrado em Engenharia de Sistemas Críticos e Computacionais (MESCC), obrigado ISEP.

Outro agradecimento está dedicado ao meu orientador, António Barros, que para além de ter sido um professor e orientador bastante prestável também contribuiu para esta minha etapa académica, obrigado António Barros.

Com especial carinho, agradeço por todo o apoio psicológico e não só que me deste pois todo esse apoio foi imprescindível para tornar este caminho possível. Obrigado Daniela Martins por seres o pilar em todos estes momentos.

Agradeço também a todos os meus amigos que estiveram presentes neste processo e que da sua forma, deram o seu apoio, força e conselhos. Portanto, obrigado André, obrigado Miguel, obrigado Alberto e obrigado Paulo.

Por fim, um último agradecimento à minha família que também contribuíram para este momento ser possível. Dando o apoio, conselhos e conhecimento possíveis. Portanto, obrigado Pai, obrigado Irmão e obrigado Mãe.

Conteúdo

Lista de Figuras

Lista de Tabelas

Lista de Abreviações

API	**A**pplication **P**rogramming **I**nterface
ATS	**A**utomatic **T**ransfer **S**witch
AWS	**A**mazon **W**eb **S**ervice
CPU	**C**entral **P**rocessing **U**nit
CoAP	**Co**nstrained **A**pplication **P**rotocol
CO	**C** **M**onoxide
CORS	**C**ross **O**rigin **R**esource **S**haring
DOS	**D**enial **O**f **S**ervice
DTLS	**D**atagram **T**ransport **L**ayer **S**ecurity
EUA	**E**stados **U**nidos da **A**mérica
FTP	**F**ile **T**ransfer **P**rotocol
GPIO	**G**eneral **P**urpose **I**nput/**O**utput
HTTP	**H**yper**T**ext **T**ransfer **P**rotocol
IACAM	**I**ndoor **A**ir **C**ontrol **A**nd **M**onitoring
IA	**I**ntellegence **A**rtificial
IPMA	**I**nstituto **P**ortuguês do **M**ar e **A**tmosfera
IP	**I**nternet **P**rotocol
ISR	**I**nterrupt **S**ervice **R**outine
IT	**I**nformation **T**echnology
IoT	**I**nternet **o**f **T**hings
JSON	**J**avaScript **O**bject **N**otation
LED	**L**ight **E**mitting **D**iode
LOB	**L**ine **O**f **B**usiness
LTS	**L**ong **T**erm **S**upport
M2M	**M**achine **T**o **M**achine
MQTT	**M**essage **Q**ueuing **T**elemetry **T**ransport
MitM	**M**an **i**n **t**he **M**iddle
OEE	**O**verall **E**quipment **E**ffectiveness
P&D	**P**esquisa e **D**esenvolvimento
PoE	**P**ower **O**ver **E**thernet
QOS	**Q**uality **O**f **S**ervice
RPO	**R**ecovery **P**oint **O**bjective
RTO	**R**ecovery **T**ime **O**bjective
SSH	**S**ecure **S**ocket **S**hell
SSL	**S**ecure **S**ockets **L**ayer
TCP	**T**ransmission **C**ontrol **P**rotocol
TLS	**T**ransport **L**ayer **S**ecurity
UDP	**U**ser **D**atagram **P**rotocol
UDP	**U**ser **D**atagram **P**rotocol
UPS	**U**ninterruptible **P**ower **S**upply
URI	**U**niform **R**esource **I**dentifier

W3C **W**orld **W**ide **W**eb **C**onsortium

Capítulo 1

Introdução

Desde o seu início, na década de 1950, o impacto das Tecnologias da Informação (TI) na economia vem crescendo significativamente. Porém, quando em 1995 a Internet totalmente aberta ao público, as TI passaram a desempenhar um papel fundamental nos negócios. Hoje em dia, a maioria das empresas conta com as TI como um ativo crítico na sua operação, de tal modo que qualquer falha resultará em perdas consideráveis.

Com o aumento do fluxo de informação e dados, também o número de centros de dados (*data centres*) e salas de servidores (*server rooms*) teve um forte crescimento. Os *data centres* concentram os recursos de computação, comunicações e equipamentos associados, podendo ocupar uma parte de um edifício, a totalidade de um edifício, ou até mesmo distribuir-se por um conjunto de edifícios mais ou menos dispersos geograficamente.

Estas infraestruturas armazenam informações críticas como, por exemplo, dados relacionados a pessoas singulares, dados de empresas, dados governamentais, entre outros. Portanto, é de esperar que as operações das TI se tornem cruciais para o enriquecimento e a continuidade dos negócios.

No entanto, ao projetar esta infraestrutura, é necessário considerar atentamente as respetivas particularidades do projeto. A **localização** tem como objetivo analisar onde é mais viável criar ou atribuir salas de servidores dentro das instalações da organização. Para isto é necessário ter em consideração diversos dados técnicos e arquitetónicos. É importante evitar espaços com janelas, evitar paredes externas e evitar locais suscetíveis a interferências. O **ar condicionado** é utilizado para controlar a temperatura e humidade dentro de uma sala de servidores, sendo uma tecnologia fundamental para a fiabilidade da operação das TI.

Por fim, é também importante a **proteção contra incêndios**, onde o principal objetivo é detetar e alertar para um incêndio na sua fase inicial e controlá-lo sem afetar o bom funcionamento do fluxo empresarial, evitando também uma ameaça para os trabalhadores.

Com o desenvolvimento da tecnologia e dos media digitais, a transformação digital tornou-se essencial, podendo ser impulsionada pelas necessidades do setor ou funcionais da empresa, ou pelas necessidades dos líderes de linha de negócios (*Line of Business* – LOB). Com a transformação digital têm surgido tecnologias e ferramentas que atendem várias necessidades, tais como arquitetura híbrida de rede (*hybrid cloud architecture*), análise profunda (*deep analytics*), inteligência artificial (*artificial intelligence* – AI), *blockchain*, automação, *edge computing* e *Internet of Things* (IoT), entre outras. (Meier 2022)
Todas estas tecnologias suportam e promovem a transformação digital dos negócios, mas suportam-se intensamente nos recursos das TI.

Por exemplo, a *hybrid cloud architecture* usa recursos localizados tanto na rede local como na nuvem. O administrador pode utilizar a *hybrid cloud* como um caminho para migrar os negócios para a *cloud* ou então integrar plataformas e serviços da *cloud* à infraestrutura local existente. Pode também manter localmente dados e processos sensíveis, quer por opção, quer por força da legislação.

Um exemplo de um cenário *hybrid cloud architecture* da *Microsoft* está representado na Figura 1.1. (Microsoft 2019)(Neenan e Bigelow 2023)

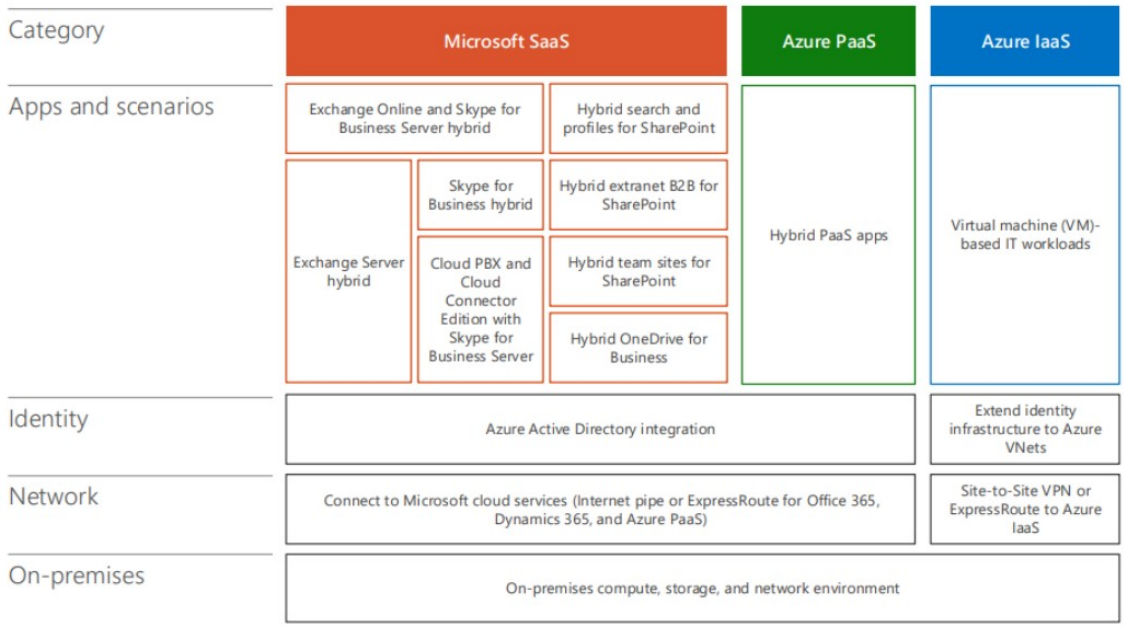


Figura 1.1: *hybrid cloud architecture*

Conforme a IBM 2023c, a mesma refere que outras tecnologias que também consomem intensivamente recursos das TI são o *blockchain* que é utilizado para garantir a integridade e inviolabilidade de dados e registos de processos e a AI que encontra cada vez mais campos de aplicação. (Wang et al. 2022)

Um exemplo prático que ilustra este cenário é o da empresa Reckitt, que tem como foco produtos de higiene, saúde e nutrição por meio de marcas como Lysol, Air Wick, Calgon, entre outras. A Reckitt fez parceria com a IBM para implementar tecnologias avançadas para conectar e digitalizar as suas fábricas de acordo com os princípios da iniciativa Indústria 4.0. O foco principal dessa digitalização foi o de fornecer aos seus funcionários as informações necessárias para prever e melhorar a eficácia geral do equipamento (OEE), a eficiência energética e a gestão da manutenção.

Desta forma se compreende que as TI têm um papel crucial para muitas organizações. No entanto, é importante notar que as TI também estão sujeitas a vários riscos que podem comprometer desde a integridade lógica (com foco riscos de cibersegurança), até à integridade física dos equipamentos, sendo este último o principal objetivo do desenvolvimento deste projeto.

Como os acidentes físicos são inerentemente imprevisíveis, é necessário implementar uma metodologia de *recuperação de desastres*. O termo recuperação de desastres refere-se a todo o processo que envolve a manutenção de uma ou mais infraestruturas e sistemas

essenciais ao bom funcionamento de uma empresa, após um acidente e desastre de origem natural ou não-natural. Neste trabalho, o processo de recuperação de desastres é orientado às tecnologias de informação que suportam funções críticas. Este processo encontra-se ilustrado na Figura 1.2.

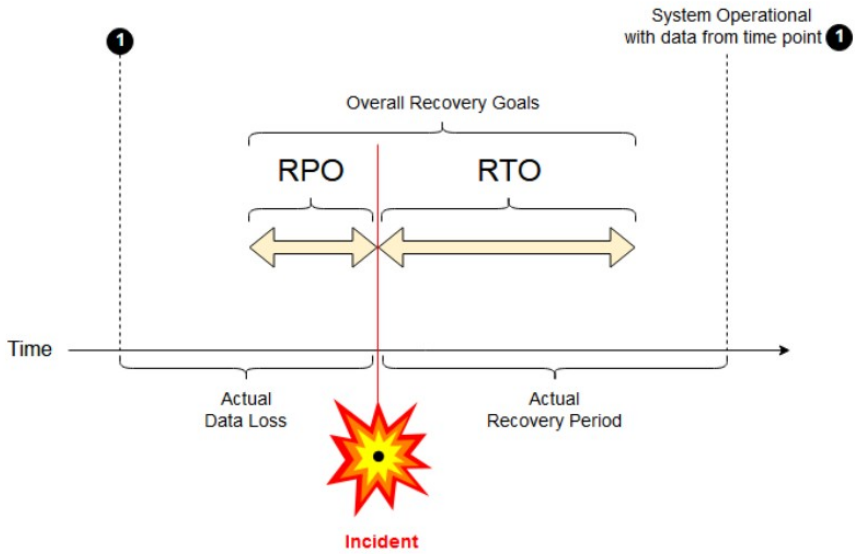


Figura 1.2: Processo de recuperação de desastres: Recovery Point Objective e Recovery Time Objective

O termo *Recovery Point Objective* (RPO), refere-se ao intervalo máximo aceitável durante o qual os dados transacionais são perdidos de um serviço das TI. Por outro lado, o termo *Recovery Time Objective* (RTO), está relacionado com a duração de tempo pretendida e o nível de serviço dentro do qual um processo de negócios deve ser restaurado após uma interrupção para evitar uma quebra na continuidade dos negócios (IBM 2023a, IBM 2023b).

1.1 Declaração do problema

Os recursos de TI são recursos cruciais a muitas organizações, mas estão sempre sujeitos a acidentes cujas consequências podem ser muito gravosas: alto risco de perda permanente de dados individuais ou corporativos, ou até mesmo falha total dos sistemas.

Desta forma, a criação ou implementação de sistemas de prevenção de incêndios deve ser uma das medidas imprescindíveis para qualquer empresa com este tipo de infraestruturas.

No entanto, esses sistemas de proteção contra incêndio devem ser projetados levando em consideração o cenário individual de cada empresa, uma vez que as condições de espaço, o sistema de ventilação, entre outras características variam de empresa para empresa. (Wickel 2008)

Esta tese debruça-se sobre a utilização tecnologias de código-aberto para desenvolvimento de sistemas de monitorização e controlo das condições ambientais em salas de servidores para prevenção e deteção precoce de acidentes.

1.2 Relevância do assunto

As TI são definidas pelo dicionário de inglês da Cambridge (English 2023) como "a ciência e a atividade de usar computadores e outros equipamentos eletrónicos para armazenar e enviar informações". As TI são utilizadas em praticamente todos os campos da atividade humana.

Por exemplo, na saúde pública têm um papel importante na realização de estatísticas vitais, investigação e pesquisa, vigilância epidemiológica e meios de diagnóstico, entre outros (*Information Technology* 2018).

Também na educação, as TI auxiliam o processo educativo tornando-o mais eficaz e produtivo. Exemplo disso é o surgimento de plataformas de ensino à distância que permitem aos estudantes aprenderem nas suas próprias casas.

Mais ainda se poderia dizer relativamente ao impacto das TI a nível político e social.

No campo económico, o progresso tecnológico e a maturidade organizacional contribuíram para o aumento da produção, acumulação de capital e a criação de uma intensa competição entre empresas. Em resposta a essa competição, a inovação é um fator que distingue organizações, conferindo vantagem sobre os concorrente. A economia mundial está a sofrer uma rápida transformação causada pela a Internet e pelas tecnologias móveis, reinventando as indústrias tradicionais.

É, pois, natural que a proteção e salvaguarda de um investimento substancial num recurso que é crítico uma preocupação das organizações.

1.3 Meta e objetivos do trabalho

O principal objetivo deste projeto é o de desenvolver um sistema de monitorização e controlo da climatização de um centro informático, recorrendo a tecnologias de código-aberto, que possa ser facilmente adaptável a diversas morfologias de salas de servidores (*server rooms*).

Este, sistema utiliza um sub-sistema de sensores (por exemplo, sensores de temperatura, sensores de humidade) para a análise das condições físicas do espaço monitorizado e um sistema de atuadores (por exemplo, ventoinhas, ar condicionado, janelas mecânicas) para regular as propriedades físicas do espaço.

Além da monitorização do espaço físico, é possível obter uma monitorização mais pormenorizada do estado dos servidores, por meio da amostragem de dados como, por exemplo, a temperatura e a utilização de cada *Central Processing Unit* (CPU). Esses dados são, posteriormente, processados pelo sistema que, após a análise dos mesmos, decide qual a forma mais adequada de atuação para cada situação.

Adicionalmente, é possível utilizar APIs de terceiros (por exemplo, do Instituto Português do Mar e da Atmosfera - IPMA) que fornecem dados sobre a meteorologia local. Esses dados têm o potencial de adicionar informações ao sistema, o que resulta numa melhoria do seu desempenho.

Como forma de complementar este projeto, um dos parâmetros passa por desenvolver uma plataforma ou aplicação web que possibilite a configuração do sistema através de funcionalidades básicas como adicionar, editar e remover componentes. Permite também efetuar

a monitorização e controlo da climatização de um centro informático e a visualização da temperatura dentro do rack e dos seus componentes discretos como, por exemplo, CPU.

1.4 Estrutura da dissertação

Esta dissertação encontra-se organizada da seguinte forma.

No Capítulo 2, é descrito o estado da arte onde é feita uma introdução dos conceitos tendo o principal foco na importância das TI, apresentando depois as abordagens já existentes nos dias de hoje. Também são identificadas todas as tecnologias utilizadas no desenvolvimento do projeto. Findando assim esse capitulo com uma breve análise crítica.

Já no Capitulo 3 o objetivo é descrever a abordagem proposta. Para isso, é necessário apresentar a metodologia e a arquitetura que o sistema Indoor Air Controlling and Monitoring (IACAM) apresenta. Também é fulcral apresentar uma solução num caso de falha portanto, que está descrita no tópico reação após falha seguido de uma solução e uma possível arquitetura.

O Capítulo 4 relata o desenvolvimento da solução. Este é dividido em três fases sendo elas o trabalho realizado, os problemas encontrados e as alterações referentes ao plano inicial. No trabalho realizado é descrito todo o desenvolvimento, começando pelo desenvolvimento do *firmware*, a concepção do mesmo, toda a sua constituição sensorial e o protótipo desenvolvido durante este processo. De seguida, também é relatado o desenvolvimento da aplicação web e da aplicação FIWARE. O desenvolvimento do *scripting* também está detalhado neste capítulo e, por fim, é descrito o desenvolvimento do *cronjob*. Nos problemas encontrados, a finalidade passa pela enumeração de todos os problemas encontrados no desenvolvimento do projeto. De seguida, são apresentadas as soluções encontradas para os problemas que, dessa forma, ficaram resolvidos como também os problemas que não foi possível encontrar uma solução. Por fim, são mencionadas todas as alterações que foram surgindo no desenvolvimento do projeto.

No Capitulo 5 são apontadas possíveis direções de trabalho futuro, desde melhorias ao trabalho que foi desenvolvido, assim como a novas funcionalidades que poderiam ser implementadas.

Por fim, o Capitulo 6 apresenta a conclusão sobre o trabalho.

Capítulo 2

Estado da arte

Para uma organização, os equipamentos de TI representam um recurso frequentemente crítico para a sua atividade; adicionalmente, são também o produto de um investimento com custo considerável. Por estas razões, a segurança e a fiabilidade destes equipamentos é uma disciplina que é levada muito a sério, sobretudo quanto maior o valor destes recursos.

A monitorização e controlo das condições ambientais de operação é uma atividade bem estabelecida ao longo das décadas, que permite salvaguardar os equipamentos de avarias devidas à operação fora da zona nominal de operação. Muito embora o controlo das condições ambientais seja um fator que promove a segurança dos equipamentos, não é por si só suficiente, pois estes estão sempre sujeitos à possibilidade de acidentes imprevistos que poderão resultar em danos.

Desta forma, a proteção das instalações onde se situam os equipamentos de TI é uma atividade realizada por especialistas, responsáveis pelo projeto dos sistemas de proteção (com escolha dos equipamentos de monitorização e atuação) e também pela sua operação e manutenção.

Neste capítulo são apresentados, a título de exemplo, dois casos de aplicação de proteção contra incêndios e de monitorização de salas de servidores (Secção 2.3). Em seguida são apresentadas tecnologias com especificação pública que podem ser utilizadas para estabelecer um sistema de proteção independente de soluções proprietárias (Secção 2.4).

2.1 Proteção física de centros de dados

A manutenção das condições ambientais recomendadas para a operação dos equipamentos de TI é uma das medidas de primeira linha na proteção desses equipamentos. Neste campo, os fatores mais importantes são a temperatura e a humidade do ar.

A *American Society of Heating, Refrigerating and Air-Conditioning Engineers* (ASHRAE) é um órgão que rege o padrão para os limites de temperatura e humidade aceites para ambientes de data centres. O comité técnico da ASHRAE 9.9 determinou em 2011 que um data centre Classe A deveria manter a temperatura entre 15 a 32 graus Celsius e a humidade relativa entre 20% a 80% (Vertiv 2023), conforme o diagrama representado na Figura 2.1.

Seguindo os fundamentos da ASHRAE, a IBM apresenta dois exemplos *Critérios de Design Ambiental* 2021 que representam um caso de um ambiente operacional, representado na Figura 2.2), e o de um ambiente não-operacional, representado na Figura 2.3).

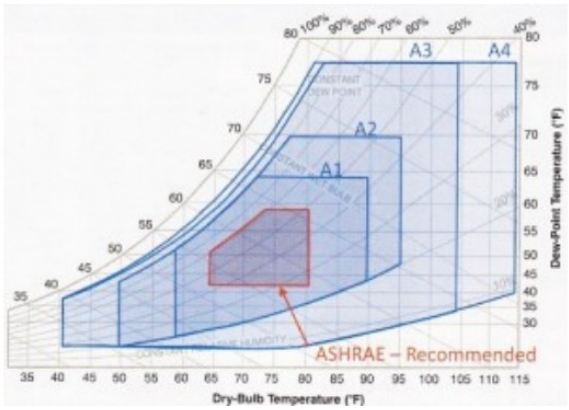


Figura 2.1: Padrões segundo ASHRAE

Tabela 1. Ambiente Operacional[1, 5]

Temperatura	18°C (64,4°F) a 27°C (80,6°F)[4]
Umidade mínima	Ponto de condensação 5,5°C (41,9°F)
Umidade máxima	Umidade relativa de 60% ou ponto de condensação de 15°C (59°F)
Contaminação gasosa	O nível de severidade G1 conforme o ANSI/ISA 71.04-1985[2], que estabelece que a taxa de reatividade dos cupons de cobre deve ser inferior a 300 Angstroms por mês (Å/mês, ≈ 0,0039 µg/cm2-ganho de peso por hora).[6] Além disso, a taxa de reatividade de cupons de prata deve ser inferior a 300Å/mês (≈ 0,0035 µg/cm^2-ganho de peso por hora).[7] O monitoramento reativo de corrosividade gasosa deve ser conduzido aproximadamente a 5 cm (2 pol.) na frente do rack no lado de entrada do ar a um quarto e três quartos da altura da estrutura do piso ou onde a velocidade do ar for muito maior.
Contaminação Particulada	Os datacenters devem atender ao nível de limpeza de ISO 14644-1 classe 8. Para datacenters sem o economizador de ar, a limpeza ISO 14644-1 classe 8 pode ser atendida simplesmente pela opção da filtragem a seguir: – O ar ambiente deve ser continuamente filtrado com filtros MERV 8. – O ar que entra em um datacenter pode ser filtrado com filtros MERV 11 ou preferencialmente MERV 13. Para datacenters com economizadores de ar, a opção de filtros para atingir limpeza ISO classe 8 depende das condições específicas presentes nesse datacenter. A umidade relativa deliquescente da contaminação por partículas deve ser superior a 60% de RH.[3] Os datacenters devem estar sem rebarbas de zinco.[8]

Figura 2.2: Ambiente operacional

Tabela 2. Ambiente Não Operacional[2]

Temperatura	5°C (41°F) a 45°C (113°F)
Umidade relativa	8% a 80%
Ponto de condensação	Menos de 27°C (81°F)
Contaminação gasosa	O nível de severidade G1 conforme ANSI/ISA 71.04-1985[1], que indica que a taxa de reatividade dos cupons de cobre deve ser inferior a 300 Angstroms por mês (Å/mês, ≈ 0,0039 µg/cm^2-ganho de peso por hora).[3] Além disso, a taxa de reatividade de cupons de prata deve ser inferior a 300Å/mês (≈ 0,0035 µg/cm^2-ganho de peso por hora).[4] O monitoramento reativo de corrosividade gasosa deve ser conduzido aproximadamente a 2 pol. (5 cm) na frente do rack no lado de entrada do ar a um quarto e três quartos da altura da estrutura do piso ou onde a velocidade do ar for muito maior.

Figura 2.3: Ambiente não operacional

As diretrizes da ASHRAE são alvo de uma constante evolução dos requisitos e limitações, com sucessivas atualizações nos anos de 2004, 2008 e 2011. Um exemplo da adaptação a novos requisitos é o da ampliação dos limites da humidade relativa para uma maior eficiência dos data centres, em 2015 Sverdlik 2015.

A não observação dos parâmetros de operação aumenta a probabilidade de falhas dos equipamentos, e consequentes falhas de serviço. Uma situação paradigmática foi a que ocorreu num data centre da Microsoft em 2013, em que os serviços Outlook e de SkyDrive ficaram inesperadamente inacessíveis por falha de vários servidores associados a estes serviços (Jones 2013).

2.2 Proteção contra incêndios

Contudo, alguns incidentes tomar proporções catastróficas, resultando no aparecimento de incêndios com danos permanentes nos equipamentos e até nas instalações.Incêndios em ambientes onde estão presentes componentes elétricos necessitam de medidas apropriadas, pois a utilização de água ou materiais líquidos pode resultar em curto circuitos e, consequentemente, piorar a situação em si. A solução habitualmente utilizada passa pelo uso de gases dielétricos, com capacidade de extinguir a combustão. As soluções mais clássicas passam pelo uso de dióxido de carbono ou de gases inertes, mas têm sido substituídos por novos gases desenvolvidos especificamente para este propósito.

Apesar disso é necessário lembrar que o uso de gases no combate ao fogo provoca uma aumento de pressão. Logo, é fundamental apresentar mecanismos de alívio de pressão para estes casos para que a área não fique comprometida. Contudo estes gases atuam de duas formas distintas, num aspeto físico e num aspeto químico. No que diz respeito ao aspeto físico, as moléculas do gás retiram energia térmica do fogo e, consequentemente, o fogo diminui. Por outro lado, no aspeto químico as moléculas tendem a dividir-se devido a essa absorção de energia e combinam-se com o oxigénio, interrompendo assim o processo de combustão.

A Figura 2.4 apresenta a comparação entre os produtos como Novee 1230, FM-200, gases inertes e dióxido de carbono (CO2) (Wickel 2008).

	Novec 1230	FM-200	Inert gases (Argon, Aragonite, Inergen, Azotes)	CO_2
Abidance in the flooded room (in case of faulty activation or fire the area always must be left)	unlimited possible	unlimited possible	temporary possible	not possible
Concentration for flooding (including 10 % fill- and extraction tolerances)	5,8%	8,4%	45…50,5%	47…57%
Residual oxygen	19,6%	19,1%	11,4…10,3%	CO_2 is toxic at >8 vol.-%
VDS-Approval	yes	yes	yes	yes
FM-Approval	yes	yes	constricted	yes
Fumigation in the fire-extinguish area	no fumigation	low fumigation	low fumigation	high fumigation
Lead time according to VDS	min. 10 sec	min. 10 sec	min. 10 sec	min. 10 sec
Lead time according to BGI	not necessary	not necessary	necessary	necessary
Time for effusion	max. 10 sec	max. 10 sec	60 to 120 sec	60 sec
Condensate formation on the pipes (after release)	no	no	yes	yes
Multi area system	limited possible	limited possible	possible	possible
Maximum pipe length	around 60m	around 60m	>> 150m	up to 150m
Danger for human	No	No	Yes	yes
Relation for over pressure relief	100%	115%	240…300%	280…360%
Required space for supply	low	low	high	middle
Environmental impact	very good	bad	good	very good
Evaluation	good procurement expensive	acceptable procurement expensive	conditionally operational procurement favourably priced	insufficient effect fatal

Figura 2.4: Comparação de gases extinção de incêndio

2.2.1 Novec 1230

Este componente atua em cenários mais exigentes, incluindo salas de controlo, centros de dados, museus, entre outros. Com o objetivo de ajudar a proteger os ativos e pessoas. Este agente suporta a continuidade operacional em caso de incêndio e ajuda a minimizar o tempo de inatividade para recuperação (3M 2023).

Este agente oferece as seguintes características:

- É capaz de extinguir um incêndio em questão de segundos e muito antes dos sistemas baseados em água ou com as descargas dos sistemas de gás inerte.
- Ajuda a proteger ativos de valor como sistemas eletrónicos, documentos em papel, artefactos e arquivos insubstituíveis. O agente Novec 1230 não se trata de um condutor de eletricidade.

- Inclui uma garantia ambiental global, a Garantia 3M™ Blue Sky℠, projetada para que o cliente se sinta mais tranquilo.
- Este agente é armazenado na forma líquida e descarregado como um gás, por isso requer aproximadamente 80% menos em termos de ocupação de espaço em comparação com os sistemas de gás inerte.
- É adequado para os perigos de incêndio das Classes A, B e C.
- Reconhecido por normas internacionais como NFPA 2001 e EN-15004

2.2.2 FM-200

O agente FM-200 foi desenvolvido pela empresa americana Chemours, tratando-se de um agente da família dos halogenados. Os sistemas que utilizam o agente FM-200 carecem de uma verificação periódica de controlo de fugas (P2i 2019).

O agente FM-200 é capaz de extinguir o incêndio de duas formas:

- uma forma física (80%) visto que produz uma mistura de ar/agente extintor com uma capacidade calorífica muito superior ao ar, e
- uma forma química (20%) ao intervir na reação em cadeia do fogo, absorvendo o calor das chamas ao mesmo tempo que provoca uma reação endotérmica.

O FM-200 permite uma rápida extinção de incêndios de classe A, B e C, com uma descarga em 10 segundos, minimizando os danos provocados pelo incêndio. O FM-200 é armazenado na fase líquida (agente gasoso liquefeito), aproximadamente a 42 bar a 20º C .

Este componente atua em aplicações típicas como, data centres, sala de bastidores/servidores, centros de comunicações, museus e arquivos, bibliotecas, reservas de arte, cofres, sala de transformadores, sala de quadros elétricos, sala de UPS/baterias e salas de geradores.

2.3 Abordagens existentes

A monitorização e proteção das salas de servidores são uma atividade estabelecida por agentes especialistas. São apresentadas duas abordagens disponíveis no mercado, com o intuito de representar o que é frequentemente realizado neste domínio, bem como termos de comparação com o projeto desenvolvido no âmbito desta dissertação: a *Fire extinguisher systems in the data center*, desenvolvida pela empresa alemã Rittal GmbH & Co. KG e a *Server Rack Monitoring System* desenvolvida pela empresa AKCP, dos Estados Unidos da América (EUA)

2.3.1 Fire extinguisher systems in the data centre - Rittal GmbH & Co. KG

Relativamente ao primeiro trabalho relacionado *Fire extinguisher systems in the data center*, foi estudado e desenvolvido pela empresa alemã Rittal GmbH & Co. KG, cujo seu principal objetivo é desenvolver um sistema de deteção de fumo e incêndio em centros de dados utilizando sensores e atuadores de forma a atuar no sistema em caso de algum incidente.

Este projeto leva em consideração os possíveis gases que podem ser produzidos e libertados durante um incêndio no data center e compara diferentes gases de extinção de incêndio, ou seja, gases que usados diretamente no foco do incêndio têm uma resposta rápida para

a sua extinção. Em seguida, os autores criam uma ilustração de um possível data center devidamente equipado com todos os componentes necessários para prevenir e alertar caso ocorra um incêndio na infraestrutura.
Caso um incêndio se desenvolva, o data center possui todos os equipamentos necessários para extinguir o próprio incêndio (Wickel 2008).

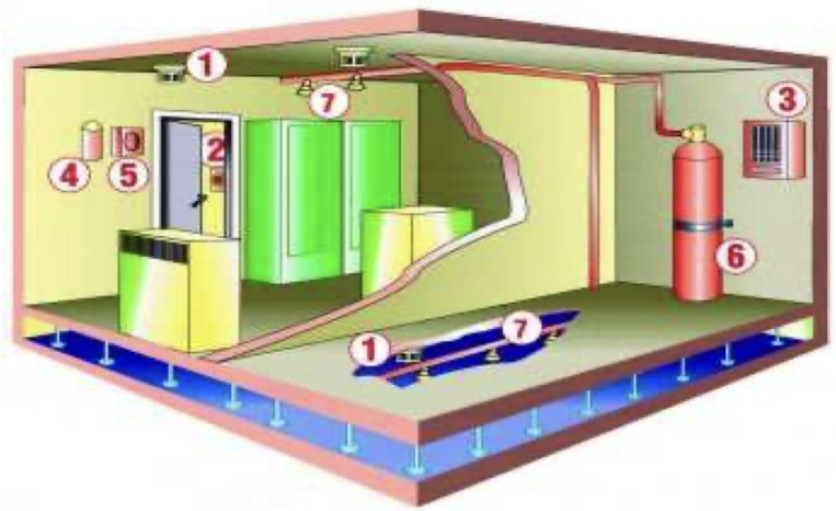


Figura 2.5: Data center with room extinguisher system

2.3.2 Server Rack Monitoring System - AKCP

Este segundo trabalho relacionado está mais centrado na importância dos sistemas de controlo nos bastidores, identificando os tipos de controlos e, finalmente, aconselhando sobre como escolher esses mesmos sistemas de monitorização.

A empresa AKCP, afiliada nos EUA, inicialmente destaca a estrutura de um Sistema de Monitorização de rack de Servidores e, depois desta primeira abordagem, realça a sua importância.

Logo de seguida, a empresa identifica os tipos de ferramentas de monitorização dos racks, onde apresentam inúmeros sensores, como, por exemplo, sensores de temperatura, sensores de humidade, sensores de pressão de ar, sensores de contacto, sensores de vibração e monitorização de energia (AKCP 2021).

Como forma de finalizar, a AKCP aconselha como escolher as ferramentas de monitorização de servidor corretas, falando também sobre o seu produto AKCPro Server, citando AKCP 2023 *"AKCPro Server is a best in class Data Center infrastructure Management Software (DCIM). Monitor all your deployed sensors, intelligent PDU's, backup power systems from a single user interface. 3D data center visualization, rack mapping, capacity planning, asset tracking and more..."*.

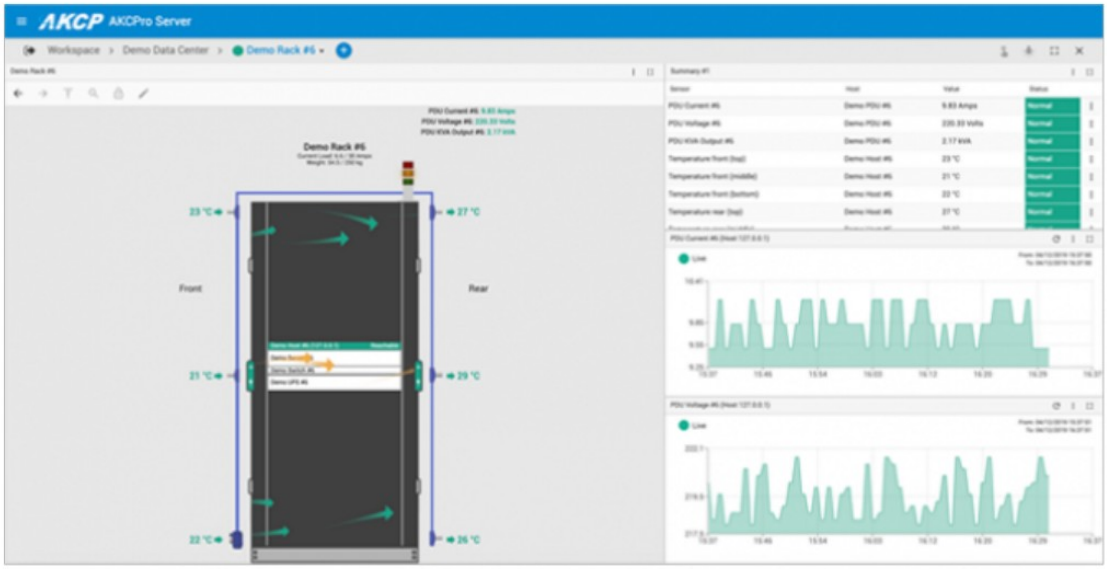


Figura 2.6: AKCPro Server

2.4 Tecnologias

De seguida, são abordadas as tecnologias disponíveis para a construção da solução para o problema proposto, partindo da visão da arquitetura do sistema.

A Figura 2.7 representa a visão da arquitetura do sistema IACAM, com a identificação dos seus componentes e respetivas ligações por onde fluem os dados.

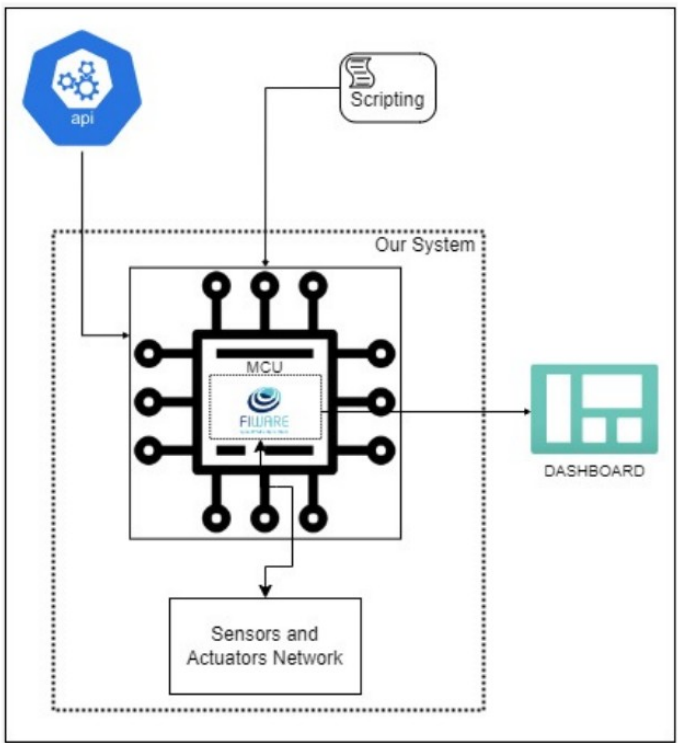


Figura 2.7: Arquitetura simplificada do IACAM

2.4.1 Servidor principal

Relativamente ao servidor principal, o mesmo tem como finalidade suportar a tecnologia FIWARE, suportar a aplicação web, função de base de dados, receber informações e dados dos servidores presentes no data center e, por fim, realizar requisições GET à API de terceiros (IPMA).

Quanto às características do servidor principal, podem ser verificadas:

- Processador - AMD Ryzen 5 5600X 6-Core 4.6GHz
- Memória - 6 GB
- Processadores - 4
- Armazenamento - 50 GB
- Sistema Operativo - Ubuntu server
- Rede - Bridged Adapter

2.4.2 Controladores embebidos

Nesta secção, são abordadas as placas que tornam possível o desenvolvimento do *firmware*. Foram consideradas duas opções diferentes e discutidas as vantagens e desvantagens da utilização das mesmas. Por fim, foi selecionada a mais vantajosa tendo por base as características deste projeto.

Raspberry Pi 4 Model B

O Raspberry Pi 4 Modelo B é um computador de placa única desenvolvido pela Raspberry Pi Foundation. É a quarta geração de computadores Raspberry Pi, sucedendo o Raspberry Pi 3 Modelo B+.

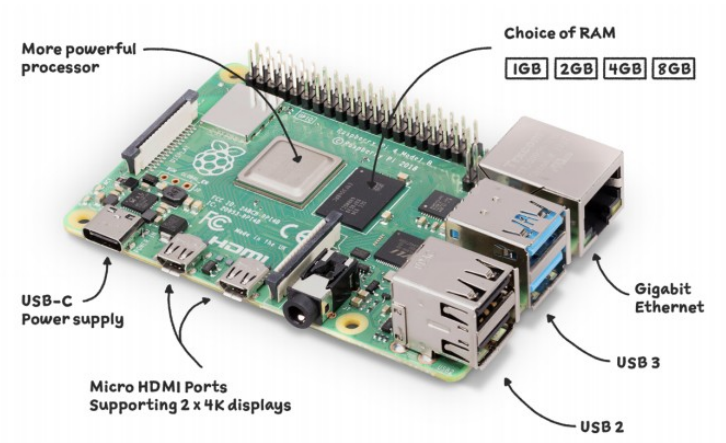


Figura 2.8: Raspberry Pi 4 modelo B, Pi 2023

O Raspberry Pi 4 Modelo B possui um SoC Broadcom BCM2711, quad-core Cortex-A72 (ARM v8) de 64 bits a 1,5 GHz e vem com 2 GB, 4 GB ou 8 GB LPDDR4-3200 SDRAM, no modelo. Possui wireless 802.11ac de banda dupla, Bluetooth 5.0, BLE, Gigabit Ethernet, duas portas USB 3.0, duas portas USB 2.0 e duas portas micro-HDMI (até 4Kp60 suportadas), bem como um conector de áudio de 3,5 mm e um cabeçalho GPIO de 40 pinos. O

dispositivo pode ser alimentado por um conector USB-C, fornecendo até 3A de energia e também suporta Power-over-Ethernet (PoE) com o PoE HAT.

O Raspberry Pi 4 Modelo B é executado numa variedade de sistemas operativos, incluindo o Raspberry Pi OS (baseado no Debian Linux) e oferece suporte a uma ampla variedade de aplicações, incluindo média, consola de jogos, computadores de mesa e muito mais. Com o seu desempenho e recursos aprimorados, o Raspberry Pi 4 Modelo B é adequado para uma ampla gama de projetos e hobbyists.

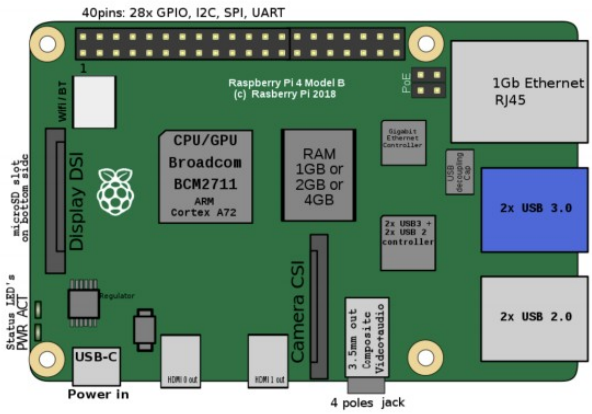


Figura 2.9: Arquitetura Raspberry Pi 4 model B, Jstrom99 2019

Microcontrolador - ESP32

Um micro controlador é um pequeno computador num chip projetado para controlar um dispositivo específico ou executar uma tarefa específica. Ele normalmente contém um processador, memória e periféricos de entrada/saída (I/O) programáveis.

Os micro controladores são usados numa ampla gama de aplicações, desde simples eletrodomésticos até sistemas muito complexos industriais.

O ESP32 é um tipo específico de micro controlador desenvolvido pela Espressif Systems. É um micro controlador dual-core de baixo consumo de energia com conectividade Wi-Fi e Bluetooth integrada. O ESP32 possui dois núcleos Xtensa LX6 de 32 bits, que podem ser usados para executar tarefas de forma independente ou em conjunto. Ele também inclui um grande número de interfaces periféricos, incluindo SPI, I2C, UART e I2S, bem como um MAC Ethernet 10/100, SDIO/SPI de alta velocidade e periféricos auxiliares.

O ESP32 suporta uma variedade de sistemas operativos, incluindo FreeRTOS e é amplamente utilizado na indústria da IoT para construir dispositivos conectados entre si.

Figura 2.10: Microcontroller ESP32, ESPRESSIF 2023

Relativamente à arquitetura do ESP32, a figura seguinte 2.11 representa o diagrama de pinos (pinout), do ESP Wroom32.

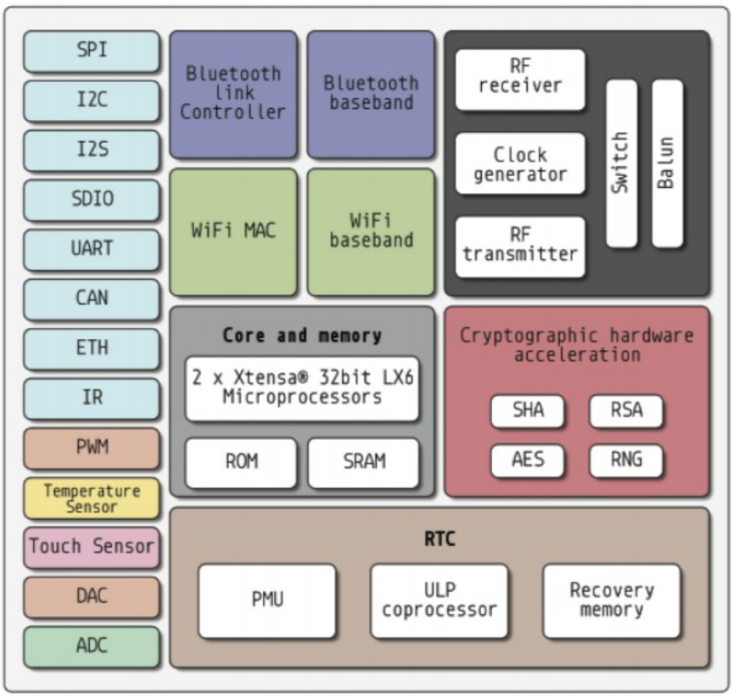


Figura 2.11: Arquitetura do ESP32, embedded 2023

2.4.3 Plataformas para gerir a Internet das Coisas

FIWARE

Relativamente a esta tecnologia, o FIWARE proporciona uma *framework* de código-aberto (*open-source*) que permite que diversos componentes sejam acoplados ou assemblados em conjunto e até com componentes de terceiros para desenvolver uma plataforma.

O principal componente e objetivo de qualquer plataforma e solução "Powered by FIWARE" é um *FIWARE Context Broker Generic Enabler*. A principal função do *Context Broker* é a necessidade de administrar informações de contexto, permitindo-lhe realizar atualizações e trazer acesso ao contexto.

Para isso, o *FIWARE Context Broker* precisa do suporte do *FIWARE NGSI* que é a API utilizada para a integração dos componentes dentro de uma plataforma "Powered by FIWARE" e de aplicações para atualizar ou consumir informações de contexto.

Construído em torno do *FIWARE Context Broker*, está disponível um amplo conjunto de *FIWARE Generic Enablers* de código aberto complementar, lidando com o seguinte:

- Interface com a IoT, robôs e sistemas de terceiros, para capturar atualizações sobre informações de contexto e traduzir as ações necessárias.
- Processamento, análise e visualização de informações de contexto que implementam o comportamento inteligente esperado de aplicações e auxiliam os utilizadores finais na tomada de decisões inteligentes.
- Dados de contexto e gestão de API, publicação e monetização de dados de contexto e API, trazendo suporte ao controlo de uso e a oportunidade de publicar e monetizar parte dos dados geridos.

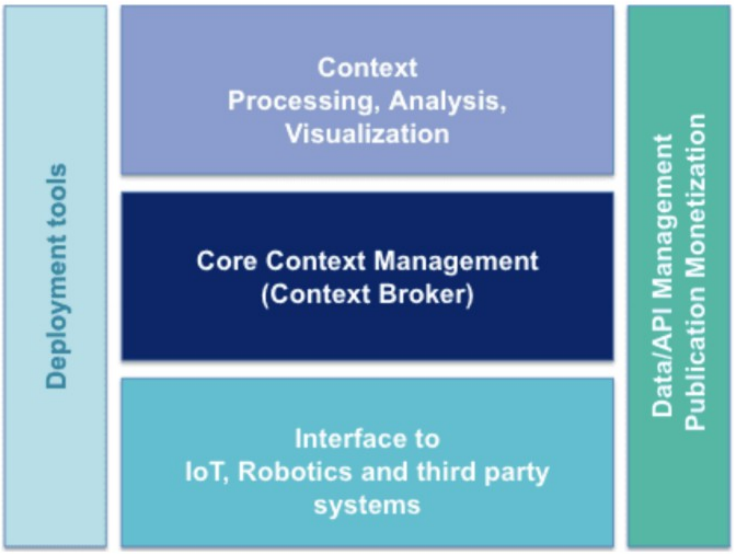


Figura 2.12: Fiware, FIWARE 2023

O principal propósito da utilização da plataforma FIWARE neste projeto é a facilidade de interação entre os sensores e atuadores com a plataforma final e auxiliar o fluxo de dados no sistema. Ou seja, o FIWARE é responsável por tratar toda a comunicação entre os dispositivos IoT, do qual fazem parte os sensores e o servidor utilizado para armazenar e exibir os dados.

A figura a seguir 2.13, representa uma possível arquitetura para o sistema IACAM. Nesta figura é possível ver o objetivo do FIWARE na aplicação, onde ele funciona como um *middleware*.

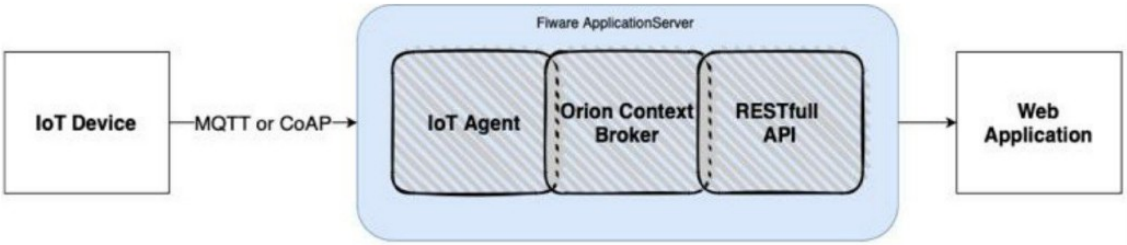


Figura 2.13: Arquitetura da aplicação Fiware

ThingSpeak

O ThingSpeak é uma plataforma de análise IoT que permite aos *developers* recolher, armazenar, analisar e visualizar dados de dispositivos IoT. Ele fornece um conjunto de APIs para gestão e análise de dados, semelhante à tecnologia FIWARE.

É também uma plataforma poderosa e flexível que fornece uma variedade de ferramentas e componentes para reunir, analisar e visualizar dados de dispositivos conectados. Os *developers* podem usar o ThingSpeak para criar aplicações IoT personalizadas para uma ampla gama de casos de uso, desde automação residencial até monitorização e controlo industrial.

Desta forma, alguns dos principais recursos e componentes da plataforma ThingSpeak passam por:

- Canais:
  - Os canais são o componente principal do ThingSpeak e fornecem uma forma de reunir e organizar dados de dispositivos conectados.
  - Cada canal possui até oito campos que podem ser usados para armazenar diferentes tipos de dados.
- APIs:
  - O ThingSpeak fornece uma variedade de APIs que os *developers* podem usar para interagir com a plataforma e enviar dados de e para dispositivos conectados. Essas APIs incluem APIs RESTful, MQTT e webhooks.
- *MATLAB Analytics*:
  - O ThingSpeak oferece integração com o MATLAB, que permite aos *developers* realizar análises avançadas e *machine learning* nos dados reunidos dos dispositivos conectados.
- Ferramentas de visualização:
  - O ThingSpeak fornece uma variedade de ferramentas de visualização, incluindo gráficos, medidores, mapas e linhas do tempo, que permitem aos *developers* criar *dashboards* personalizados para monitorizar e analisar os dados dos dispositivos conectados.
- Plugins:
  - O ThingSpeak oferece suporte a plugins que ampliam a funcionalidade da plataforma, como integração com outros serviços como X ou IFTTT.
- Aplicação móvel:

– O ThingSpeak oferece uma aplicação móvel que permite aos utilizadores monitorizar e controlar os seus dispositivos conectados a partir dos seus dispositivos móveis.

A figura 2.14, representa a arquitetura presente na plataforma ThingSpeak.

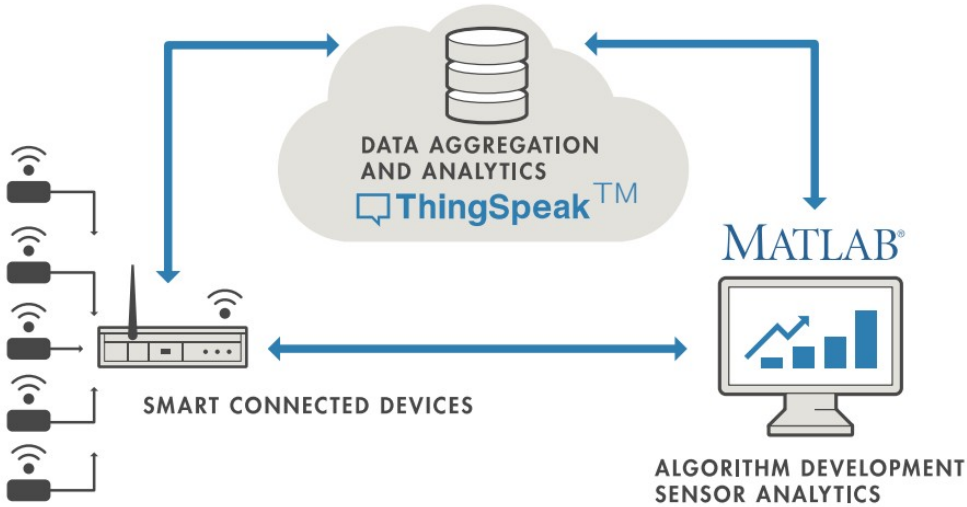


Figura 2.14: Architecture ThingSpeak Application, ThingSpeak 2023

FIWARE versus ThingSpeak

As tecnologias FIWARE e ThingSpeak são plataformas cujo o foco é gerir a Internet das Coisas. Contudo, ambas as tecnologias têm algumas diferenças em termos de recursos e capacidades.

No geral, a principal diferença entre o FIWARE e o ThingSpeak é que o primeiro é uma plataforma mais abrangente que fornece uma gama mais ampla de recursos e é mais personalizável, enquanto o ThingSpeak é uma plataforma mais focada na visualização e análise de dados. Tendo em consideração o valor monetário, o FIWARE é uma plataforma de código aberto e gratuita enquanto a plataforma ThingSpeak oferece planos gratuitos e pagos para os utilizadores. No entanto, a versão gratuita tem algumas limitações.

Com foco na abertura e customização, o FIWARE é uma plataforma de código aberto que permite aos *developers* personalizar a plataforma de acordo com suas necessidades. Em contraste, ThingSpeak é uma plataforma de código fechado sendo menos personalizável.

Sobre a recolha e a gestão, ambas as plataformas permitem que dados sejam recolhidos de dispositivos IoT, mas o FIWARE possui recursos de gestão de dados mais avançados, como normalização, armazenamento e processamento de dados. O ThingSpeak, por outro lado, concentra-se mais na visualização e análise de dados.

No ponto de vista da conectividade, a plataforma FIWARE possui uma gama mais ampla de opções de conectividade e suporta diversos protocolos, incluindo HTPP, MQTT e CoAP. O ThingSpeak, por outro lado, é mais limitado em termos de opções de conectividade.

Relativamente às APIs e integração, ambas as plataformas fornecem APIs para os *developers* integrarem as suas aplicações com a plataforma, mas o FIWARE possui uma gama mais ampla de APIs e opções de integração.

Por fim, no tópico suporte à comunidade, o FIWARE possui uma comunidade maior e mais ativa de *developers*, o que significa que há mais suporte disponível para os *developers* que encontram problemas ou precisam de ajuda nas suas aplicações e projetos.

2.4.4 Protocolos de comunicação

Constrained Application Protocol

O Constained Application Protocol (CoAP), é um protocolo leve de comunicação da camada de aplicação (application-layer), projetado especificamente para dispositivos IoT com recursos computacionais e de rede limitados. É utilizado para comunicação entre dispositivos IoT, bem como entre dispositivos IoT e servidores na Internet.

O CoAP utiliza o User Datagram Protocol (UDP), como a sua camada de transporte, que fornece comunicação rápida e com baixo *overhead*. No entanto, o UDP não fornece entrega confiável, então o CoAP inclui mecanismos para detectar e retransmitir pacotes perdidos.

Este protocolo usa um modelo de *request/response* para a comunicação, onde um dispositivo envia um *request* para outro dispositivo e recebe um *response*. Isso facilita a criação de relações cliente-servidor simples entre os dispositivos.

Sobre o CoAP, o mesmo foi desenvolvido para trabalhar com recursos, onde cada recurso é identificado por uma *Uniform Resource Identifier* (URI) única. Os dispositivos podem solicitar informações sobre recursos e os servidores podem fornecer informações sobre o estado de seus recursos.

No CoAP, as mensagens são codificadas em formato binário, que é mais eficiente do que os formatos baseados em texto e reduz a sobrecarga de comunicação.

O CoAP inclui recursos de segurança, como criptografia, usando Datagram Transport Layer Security (DTLS), e verificação da integridade da mensagem, para garantir que os dados transmitidos pela rede sejam seguros.

O CoAP foi projetado para funcionar com outros padrões de IoT, como MQTT e HTTP, facilitando a integração com sistemas e ferramentas existentes.

No geral, o CoAP fornece um protocolo de comunicação simples, eficiente e seguro para dispositivos IoT, tornando-o uma opção atraente para *developers* e organizações que desejam criar aplicações IoT.

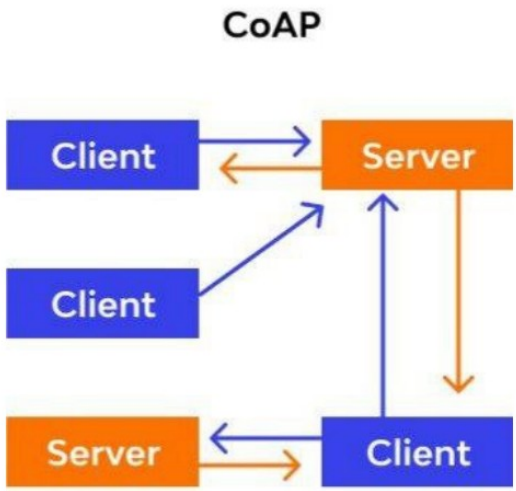


Figura 2.15: CoAP Architecture, Wallarm 2023

A figura 2.15, representa a arquitetura presente no protocolo CoAP. Conforme mencionado anteriormente, os clientes trocam dados com eles usando o servidor.

Message Queuing Telemetry Transport

O Message Queuing Telemetry Transport (MQTT), é um protocolo de mensagens baseado em *publish/subscribe* que é comummente utilizado em comunicações na Internet das Coisas e em máquina para máquina (M2M).

É um protocolo leve que utiliza uma arquitetura cliente-servidor e opera no topo da pilha TCP/IP. O servidor atua como um intermediário, que tem o propósito de gerir a distribuição de mensagens entre os clientes.

Os clientes numa rede MQTT podem publicar mensagens num determinado tópico, que se trata de uma *string* que atua como um endereço para as mensagens. Os clientes que estão subscritos recebem mensagens que são publicadas nos tópicos do seu interesse.

O protocolo define vários níveis de qualidade de serviço (QoS), para mensagens, desde QoS 0 (dispara e esquece) até QoS 2 (entrega garantida).

O protocolo MQTT também inclui diversos recursos que oferecem suporte à entrega confiável de mensagens. Desde a persistência de mensagem a mensagens *last will and testament*, que são enviados caso um cliente se desconectar de forma inesperada. E também oferece persistência de sessão, ou seja, permite que um cliente continue a partir do ponto onde parou após a desconexão.

Este protocolo também proporciona que a autenticação e criptografia podem ser adicionadas às comunicações MQTT usando os mecanismos de segurança TCP/IP padrão.

No geral, o MQTT foi projetado para ser eficiente, escalável e fácil de implementar, tornando-o uma escolha ideal para comunicações IoT e M2M, onde a largura de banda e a duração da bateria geralmente são recursos limitados.

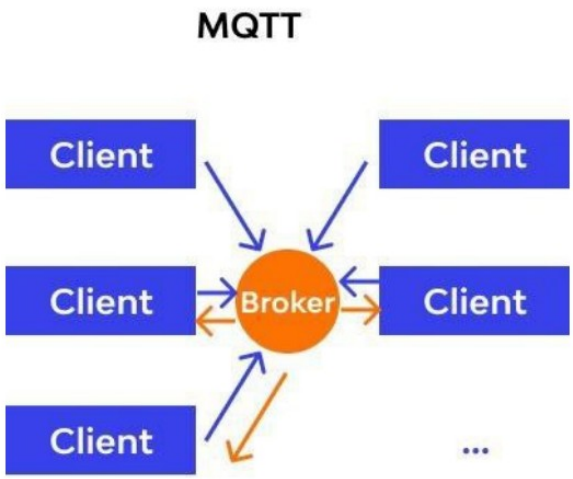


Figura 2.16: MQTT Architecture, Wallarm 2023

A figura 2.16, representa a arquitetura presente no protocolo MQTT. Conforme mencionado anteriormente, os clientes publicam ou subscrevem os tópicos presentes no *broker* e compartilham os dados com outros clientes.

CoAP versus MQTT

Como já foi mencionado, MQTT e CoAP são protocolos de comunicação projetados para dispositivos e comunicações IoT. No entanto, os mesmos apresentam diferenças tanto no objetivo como nas características de design e são utilizados em diferentes tipos de aplicações IoT.

Algumas diferenças entre os protocolos MQTT e CoAP, são:

- Propósito:
  - O MQTT foi projetado especificamente para aplicações de IoT, com foco na comunicação bidirecional confiável de baixa latência entre dispositivos. O CoAP, por outro lado, foi projetado para dispositivos com recursos limitados, com foco na simplicidade e no uso eficiente dos recursos de rede.
- Camada de transporte:
  - O MQTT usa o TCP como camada de transporte, fornecendo assim uma entrega confiável e ordenada de mensagens. Enquanto o CoAP usa UDP, que é mais rápido e requer menos sobrecarga, mas não garante uma entrega confiável.
- Formato da mensagem:
  - O MQTT usa um modelo de *publish/subscribe* para a sua comunicação, onde os dispositivos podem fazer *subscribe* a tópicos e receber mensagens de outros dispositivos que fazem *publish* nesses mesmos tópicos. O CoAP usa um modelo de *request/response*, onde um dispositivo envia um *request* para outro dispositivo e, por sua vez, recebe um *response*.
- Segurança:
  - O MQTT possui recursos de segurança integrados, incluindo criptografia SSL/-TLS, autenticação de utilizador e controlo de acesso. O CoAP também possui recursos de segurança, como criptografia DTLS, contudo estes não são tão bem estabelecidos quando comparados com os do MQTT.

Em suma, o MQTT é adequado para aplicações que exigem comunicação confiável e de baixa latência e podem lidar com a sobrecarga de um protocolo baseado em TCP. O CoAP é mais adequado para aplicações em dispositivos com recursos limitados que precisam de comunicar uns com os outros de forma rápida e eficiente e podem trocar alguma confiabilidade por uma comunicação mais rápida.

De seguida, são apresentadas algumas comparações entre os protocolos MQTT e CoAP.

MQTT	CoAP
This model has publishers and subscribers as main participants	Uses requests and responses
Central broker handles message dispatching, following the optimal publisher to client path.	Message dispatching happens on a unicasting basis (one-to-one). The process is same as HTTP.
Event-oriented operations	Viable for state transfer
Establishing a continual and long-lasting TCP connection with the broker is essential for the client.	Involved parties use UDP packets (async) for message passing and communication.
No message labeling but have to use diverse messages for different purposes.	It defines messages properly and makes its discovery easy.

Figura 2.17: MQTT versus CoAP, Wallarm 2023

2.4.5 Frameworks JavaScript para aplicações web

Vue.js

A framework Vue.js é uma framework JavaScript de software livre (open-source) para desenvolver interfaces e aplicações.
Esta framework usa uma arquitetura baseada em componentes, na qual os componentes da interface do utilizador são escritos como componentes Vue e podem ser reutilizados em toda a aplicação. Os componentes Vue são flexíveis e podem ser facilmente combinados para construir interfaces complexas.

Uma das principais características do Vue.js é o seu sistema de reatividade, que atualiza automaticamente a interface quando os dados subjacentes mudam ou sofrem alterações. Isso facilita a criação de aplicações dinâmicas e responsivas. O Vue.js também fornece um conjunto de ferramentas para gerir o estado de uma aplicação, incluindo um armazenamento centralizado para gerir os estados globais e um sistema para lidar com os estados ao nível do componente.

Angular

O Angular é também uma framework JavaScript para o desenvolvimento de aplicações web.

O Angular utiliza uma arquitetura baseada em componentes, onde esses componentes da interface são escritos como componentes Angular e podem ser combinados para criar interfaces complexas. A estrutura também fornece um conjunto de ferramentas para gerir o estado de uma aplicação, incluindo ligação de dados bidirecional, que permite atualizações contínuas na interface do utilizador com base nas alterações nos dados subjacentes.

Uma das principais características do Angular é o seu suporte ao TypeScript, que oferece, entre outras coisas, capacidades de programação orientada a objetos. O Angular também fornece um conjunto de APIs para trabalhar em conjunto com HTTP, facilitando a interação com serviços de back-end. Possui também uma comunidade grande e ativa, com muitos recursos disponíveis para iniciantes que estão a aprender e a desenvolver aplicações, tornando-se uma escolha popular para aplicações grandes e complexas e é usado por muitas empresas e organizações.

Vue.js versus Angular

O Vue.js e o Angular são ambos frameworks JavaScript populares para a criação de aplicações web. Contudo, sendo as duas *frameworks* de JavaScript apresentam, mesmo assim, diferenças quando são comparadas entre si:

- Complexidade:
  - O Angular é uma *framework* completa, com um grande número de conceitos e funcionalidades para aprender, tornando-o mais complexo quando comparado com o Vue.js.
  - O Vue.js é mais simples e fácil de aprender, tornando-o uma boa escolha para projetos mais pequenos e menos complexos ou para programadores que estão a iniciar a sua carreira na área.
- Performance:
  - O Vue.js é conhecido pelo seu desempenho leve e rápido, tornando-o numa boa escolha para aplicações com grandes quantidades de dados ou interações complexas do utilizador.
  - O Angular tem uma curva de aprendizagem mais acentuada e pode apresentar um desempenho mais lento, especialmente para aplicações maiores.
- Documentação e comunidade:
  - O Angular tem uma comunidade maior e mais presente, com mais recursos e suporte disponíveis.
  - O Vue.js tem vindo a crescer, contudo a sua comunidade e recursos ainda são menores quando comparados com os do Angular.
- Flexibilidade:
  - O Vue.js é mais flexível e fácil de integrar com outras bibliotecas e tecnologias
  - O Angular é mais dogmático e possui um conjunto de melhores práticas definidas.

2.4.6 Docker

O Docker é uma plataforma de software que permite a criação, teste e implementação de aplicações de forma rápida. Este cria pacotes de software em unidades padrão que são denominadas de *container*, que contem tudo o que o *software* necessita para ser executado. Ao utilizar o Docker, torna-se possível implementar e escalar de forma rápida aplicações em qualquer ambiente e ter a certeza de que o código será executado. (Docker 2003)

Definição de Container

Um *container* proporciona uma forma padronizada de comprimir código, configurações e dependências da aplicação num único objeto. Os *containers* compartilham um sistema operativo instalado no servidor e são executados como processos isolados, sendo que isso permite fazer implementações rápidas, confiáveis e consistentes independentemente do ambiente.
Claramente esta tipologia carrega consigo benefícios, desde a possibilidade de ser executado em qualquer lugar, melhor utilização de recursos, isto é, os *containers* proporcionam o isolamento de processos que permite configurar o uso do CPU e da memória de forma detalhada

para melhor utilização dos recursos. Por fim, permite escalabilidade, ou seja, cada *container* é executado como um processo separado que compartilha os recursos do sistema operativo subjacente sendo que torna possível que os mesmos sejam iniciados e interrompidos facilmente.

Funcionamento do Docker

Como foi dito anteriormente, o Docker permite executar o código de forma padronizada e, também, é um sistema operativo para os *containers*. Da mesma forma que uma máquina virtual virtualiza o hardware do servidor, os *containers* virtualizam o sistema operativo do servidor.

O Docker é instalado de forma independente em cada servidor e apresenta comandos simples para o utilizador poder criar, iniciar ou interromper os *containers*.

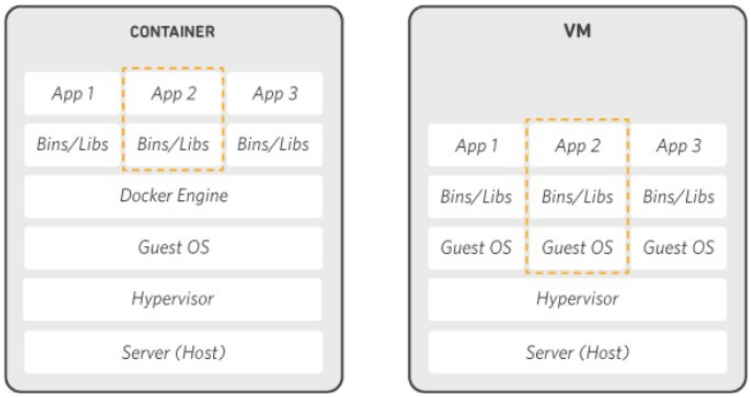


Figura 2.18: Docker

Vantagens da utilização do Docker

Ao utilizar o Docker, é possível enviar o código com maior rapidez, padronizar as operações da aplicação, mover o código com facilidade e economizar, melhorando assim a utilização dos recursos. Com a utilização da plataforma Docker, o utilizador tem apenas um único objeto que pode ser executado com segurança em qualquer lugar. O Docker apresenta uma sintaxe simples e direta o que proporciona o controlo total. A sua ampla adesão significa que existe um ecossistema robusto tanto de ferramenta como aplicações prontas para ser utilizadas em conjunto com o Docker.

- Disponibilizar mais software com maior rapidez:
  - Os utilizadores da plataforma Docker disponibilizam software com uma frequência de sete vezes maior do que os utilizadores de outras tecnologias. O Docker permite enviar serviços isolados sempre que necessário.
- Padronizar operações:
  - Pequenas aplicações em *containers* facilitam a implementação, a identificação de problemas e o *roll-back* para correção.
- Facilidade de movimentação:

  - Aplicações baseadas em Docker podem ser migradas de máquinas locais de desenvolvimento para implementações de produção, como por exemplo, na Amazon Web Services (AWS).

- Economizar custos:
  - Os *containers* do Docker facilitam a execução de mais código em cada servidor, melhorando a utilização e economizando recursos.

A utilização dos *containers* da plataforma Docker têm a possibilidade de ser um alicerce, criando aplicações e plataformas modernas. O Docker facilita a criação e, posteriormente, execução de arquiteturas de micro-serviços distribuídos, implementando o código com *pipelines* de integração. Cria, também, sistemas de processamento de dados altamente escaláveis e, por fim, plataformas totalmente geridas pelos *developers* da aplicação.

2.4.7 Ferramentas de suporte ao desenvolvimento de software

Trello

O Trello é uma ferramenta da Atlassian, cujo principal objetivo passa por auxiliar a gestão de projetos e a colaboração entre os membros do mesmo. Para isso, são utilizados quadros e sistemas de cartões com o intuito de ajudar os utilizadores e as equipas a organizar o seu trabalho. Facilitando assim, o acompanhamento de projetos, gestão de tarefas e tornar a colaboração em equipa de maneira mais flexível e intuitiva de usar.

O Trello neste projeto foi utilizado para ajudar na gestão do tempo. Para isso todas as semanas foram criadas novas etiquetas e novas tarefas que devem ser implementadas, se possível, no tempo estipulado que foi atribuído a cada uma.

GitHub

O GitHub é uma plataforma que tem o propósito de servir como um sistema de controlo de versões para repositórios de código. O seu objetivo principal é facilitar o desenvolvimento de software, fornecendo assim ferramentas para o controlo de versões, possibilidade de compartilhar código entre pessoas e equipas, *tracking* de problemas e gerir projetos.

O objetivo da utilização da ferramenta GitHub no projeto foi, como já referido nos objetivos, o controlo de versões, manter e fazer backups de todo o projeto e compartilhar informações e dados do projeto com todos os envolvidos.

Arduino IDE

A aplicação Arduino IDE tem o objetivo de ser usado para escrever *firmware* e fazer upload em placas. É um editor moderno e apresenta uma interface responsiva. Conta com preenchimento automático, navegação de código e apresenta um *debbuger*. (Arduino 2023)

Relativamente ao seu uso no desenvolvimento do projeto, a aplicação Arduino IDE foi essencial pois deu suporte na criação do *firmware*, presente no dispositivo IoT.

Visual Studio Code

A aplicação Visual Studio Code é um editor de código com suporte para operações de desenvolvimento, *debbunging*, execução de tarefas e controlo de versão. O objetivo desta aplicação fornece apenas as ferramentas que um *developer* necessita. (Microsoft 2021)

Acerca da utilização desta aplicação, a mesma teve o objetivo de dar suporte ao desenvolvimento da aplicação web, devido às suas vantagens e de fácil utilização.

2.5 Análise Critica

Após uma contextualização um pouco mais detalhada de algumas abordagens já existentes e de todas as tecnologias potencialmente cruciais para o desenvolvimento deste projeto, é fundamental fazer uma análise crítica do estado da arte.

Desta forma, face ao problema já referido no capítulo 1 e os objetivos deste trabalho e tendo em consideração que o mesmo está integrado numa área crítica, é necessário proceder à sua implementação para que seja possível criar um protótipo que corresponde ao sistema IACAM.

Para que esse desenvolvimento seja possível, foi necessário estudar e aprofundar as diferentes tecnologias que podem ser utilizadas para esse fim.

Após o estudo e análise crítica é expectável a utilização da tecnologia FIWARE, devido às suas vantagens que já foram mencionadas anteriormente. Relativamente às *frameworks* supracitadas, é mais vantajoso utilizar o Angular devido às suas mais valias e comunidade mais ativa perante a *framework* Vue.js.

No que se refere ao protocolo de comunicação, foi utilizado o MQTT no projeto para suportar a comunicação entre o dispositivo IoT e a plataforma FIWARE desenvolvida.

Quanto à *board*, infelizmente não foi possível adquirir uma Raspberry Pi 4 model B. Desta forma, a mesma foi substituída por uma máquina virtual que é detalhada posteriormente. Contudo, o micro-controlador ESP32 foi utilizado com o fim de constituir o dispositivo IoT que tem acoplado a si todos os sensores e atuadores utilizados para o desenvolvimento do projeto.

Capítulo 3

Abordagem proposta

Como forma de contextualizar o estado da arte do projeto IACAM, é necessário focar e aprofundar alguns tópicos, sendo eles respetivamente os conceitos, o design e a metodologia utilizada e os desenvolvimentos preliminares.

3.1 Conceito

As subsecções que se seguem são fundamentais para prototipar o sistema deste projeto. Através da elaboração e implementação de rascunhos da arquitetura do sistema, de fluxogramas e de diagramas, a lógica do desenvolvimento do projeto torna-se mais acessível e intuitiva. Desta forma, a identificação de possíveis falhas e erros é mais rápida e fácil.

3.2 Design e Metodologia

Como o projeto IACAM é constituído por diferentes componentes, é necessário refletir sobre qual será a melhor implementação ou arquitetura do sistema. Ou seja, no projeto, está presente a comunicação entre a *board*, os sensores e atuadores e a comunicação entre a *board* com APIs de terceiros e servidores. Estes dois tipos de comunicação são diferentes e utilizam protocolos diferentes.

Os *requests* feitos à API são feitos utilizando o protocolo HTTP onde o resultado é retornado no formato JSON.

Já a comunicação com os servidores vão ser baseadas nos protocolos SSH, FTP, HTTP, entre outros, pois ainda não há nenhum protocolo definido para esta comunicação.

A comunicação com os sensores e atuadores tem como base o protocolo MQTT.

Também é necessário ter em consideração que a plataforma FIWARE facilita o fluxo de dados entre os sensores e a aplicação web, conforme mencionado no capítulo 2. Esta é também mais acessível devido à sua própria arquitetura que passa do IoT Agent (que se preocupa com a interação de sensores e atuadores), Context Broker e RESTfull API que suporta a aplicação web.

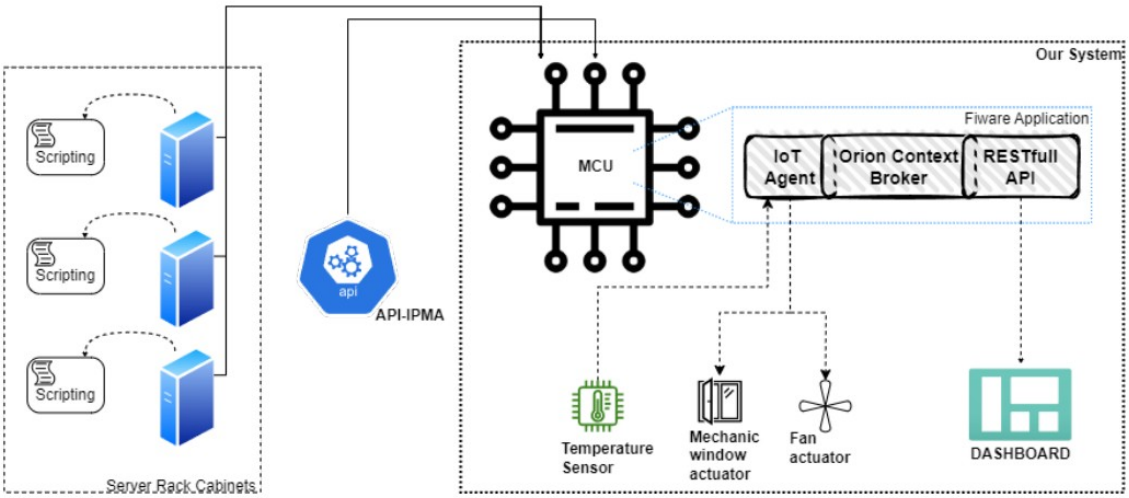


Figura 3.1: IACAM system architecture

A figura 3.1 refere-se à arquitetura primordial do sistema IACAM, que sofreu alterações posteriormente.

Após observar a imagem, é possível identificar os diferentes componentes e a lógica que será implementada no sistema IACAM. Podemos também observar onde acontece a entrada de dados de sistemas externos, APIs, servidores e os sensores e atuadores conectados à *board*.

Por fim, a implementação do FIWARE é feita diretamente na *board*. Este implementação está visivelmente identificada na figura.

3.3 Reação após falha

Relativamente a este tópico, é necessário refletir que nem sempre é possível garantir que o sistema esteja seguro ou totalmente disponível. Desta forma, é possível ocorrer situações menos favoráveis ao sistema que colocam o mesmo em causa. Estas situações podem ser, por exemplo, corte ou falha repentina de energia, ciberataques, entre outras.

Quando se projetam sistemas como este, com recurso a sensores, atuadores e canais de comunicação, é necessário pensar na resiliência e redundância do sistema. Um bom exemplo disso são os backups para garantir a operação contínua diante de falhas. A implementação de mecanismos para detetar e mitigar ataques cibernéticos ao sistema também pode ser algo a ponderar pois estes mecanismos de deteção e prevenção de intrusão ajudam bastante no tópico do mundo ciber físico. O uso de criptografia e protocolos de comunicação segura para proteger a transmissão de dados também é fundamental para uma comunicação segura entre sistemas e componentes.

Deste modo, é possível dividir essas falhas em duas tipologias distintas, as situações físicas e as ciber físicas.

Contudo para este projeto em específico e, dentro da tipologia de situações físicas, enquadra-se melhor soluções como utilização de Uninterruptible Power Supply (UPS), sistemas de backup de bateria com chave de transferência automática e fontes de alimentação redundantes com fontes de alimentação duplas.

Esta secção está dividida em duas subsecções que têm o objetivo de detalhar o que foi mencionado anteriormente e ao mesmo tempo apresentar a lógica e uma breve arquitetura de como deveriam ser implementados as soluções mencionadas.

3.3.1 Tipos de falhas

Como já mencionado anteriormente, as falhas podem ser de dois tipos distintos em que cada um apresenta situações diferentes. Sendo essas aprofundadas de seguida.

Físicas

No tópico das falhas físicas, fazem parte componentes como sensores, atuadores, energia elétrica e conectividade da rede.

Relativamente aos componentes como os sensores, existe sempre a possibilidade de, com o decorrer do tempo, aparecerem falhas associadas aos mesmos, desde falhas de leitura, ruído, entre outras. É também importante referir que cada componente tem um tempo de vida útil associado.
Entrando agora em mais detalhes, qualquer que seja o sensor, existe um desvio padrão associado na hora da leitura de dados, ou seja, existe uma mudança gradual na saída do sensor devido ao envelhecimento ou fatores ambientais, levando assim a medições imprecisas.
Também existe o ruído adjunto ao sensor, isto é, existem flutuações aleatórias nas leituras do sensor causadas por fatores externos, afetando também a qualidade dos dados.
Por fim, no que toca na componente sensores, também pode ocorrer a interferência entre sensores. Esta pode levar a que as leituras de um sensor possam ser influenciadas por um outro sensor, ocorrendo assim um *crosstalk*.

Focalizando no componente de atuadores, também existe a probabilidade de com o decorrer do tempo exista o aparecimento de falhas. Como muitas das vezes se tratam de componentes elétricos ou mecânicos, existe sempre um grau de desgaste da componente, tendo, também, cada componente um tempo de vida útil associado.
Pode ocorrer casos em que um atuador não se consiga mover ou fique preso, ou seja, o atuador, devido a problemas mecânicos ou de lubrificação pode não conseguir executar a sua função no sistema. Também pode ocorrer situações como *delay*, em que a resposta do atuador é atrasada, afetando assim o tempo e o desempenho do sistema em causa.
O atuador pode, gradualmente, exibir uma mudança no comportamento do mesmo levando assim a desvios não intencionais que, consequentemente, também podem interferir no bom funcionamento do sistema.
Por último, pode ocorrer uma "zona morta"do atuador, por outras palavras, pode ocorrer uma pequena faixa de entrada em que o atuador não consiga responder afetando assim a capacidade de resposta do sistema.

Já sobre o tópico do fornecimento de energia, o mesmo não pode ser tomado como totalmente garantido, pois podem ocorrer falhas inesperadas que comprometam a disponibilidade da mesma.
Portanto, pode ocorrer a perda repentina de energia elétrica que pode originar uma interrupção de toda a operação do sistema, causando assim tempo de inatividade e até possível perda de dados. Um outro aspeto importante a referir, pode ser as condições de baixa tensão e que podem levar ao funcionamento inadequado dos componentes, redução do desempenho e também elevar o nível de desgaste.
Além do mais, pode surgir o aparecimento de surtos de energia, ou seja, pode ocorrer picos

de tensão que podem danificar os componentes e levar a falhas imediatas ou latentes. Por outro lado, também podem ocorrer flutuações de energia, ou seja, mudanças rápidas de tensão de energia que podem afetar componentes sensíveis, resultando em erros ou comportamentos erráticos.

Para concluir o tópico das falhas físicas, falta apenas abordar a componente da conectividade da rede. Caso a mesma fique indisponível ou ocorra uma falha, pode resultar em consequências negativas no sistema.
A falha de rede, isto é, a perda de conectividade de rede, pode interromper a comunicação entre sensores, atuadores e sistemas de controlo central. Relativamente à latência e ao *jitter*, podem surgir atrasos na rede e variações nos tempo de transmissão de dados que, por sua vez, podem afetar a capacidade de resposta do sistema em tempo real.
De outro modo, também pode ocorrer a perda de pacotes, sendo que estes podem ser perdidos em trânsito podendo levar a uma troca de informações incompletas ou atrasadas. E, por fim, é importante ter em consideração o congestionamento da rede, ou seja, o alto tráfego de rede pode levar a uma comunicação mais lenta e aumentar os tempo de transferência de dados.

Ciber físicas

No que se refere a falhas ciber físicas, as mesmas têm mais ênfase no que toca a ataques cibernéticos. Contudo, este tema é muito ambíguo dado que existem diferentes categorias dentro dos ataques cibernéticos, desde ataques que comprometem o sistema a nível de disponibilidade até ataques ao nível do *firmware*. Contudo, vamos explorar alguns desses possíveis ataques de seguida.
Ataques de *Denial of Service* (DOS) são ataques conhecidos por sobrecarregar um sistema com tráfego com o intuito de torná-lo inacessível para os utilizadores ou consumidores finais do sistema.
Outro ataque que pode ser considerado neste contexto é o *Man-in-the-Middle* (MitM). Este trata-se de uma intercetação ou alteração da comunicação entre sensores, atuadores ou componentes do sistema. A injeção de dados, também é um ataque frequente que tem como objetivo injetar dados que não são verdadeiros ou dados maliciosos no sistema para conseguir manipular o comportamento do sistema em causa.
Estes ataques também podem surgir devido a vulnerabilidades de *firmware* ou software em que os atacantes exploram os pontos fracos presentes no sistema para tentar conseguir ganhar o acesso não autorizado.
Por fim, um ataque bastante simples de ser executado é a adulteração física, ou seja, o acesso físico não autorizado a componentes ou ao sistema em si, levando assim ao comprometimento ou manipulação de dados.

3.3.2 Abordagens propostas

No que toca à mitigação e resiliência das falhas, existem diversas abordagens que podem ser tomadas. Contudo, dependendo do sistema desenvolvido e das suas características, é necessário refletir e estudar quais serão as abordagens mais lógicas e favoráveis para o sistema.
Essas mesmas abordagens podem passar pela implementação de sistemas UPS para fornecer energia temporária durante interrupções, permitindo a operação contínua. Também se pode projetar sistemas com múltiplas fontes de alimentação para garantir a operação contínua em caso de falha de energia. Outro exemplo é o desenvolvimento de estratégias para priorizar

componentes críticos do sistema durante a falta de energia.
No que se refere à conectividade de rede, pode-se optar por configurações de caminhos de comunicação redundantes e utilização de mecanismos de *failover* para manter a conectividade. A qualidade de serviço também é um outro tópico que se pode ter em consideração, ou seja, implementar mecanismos de QoS para priorizar o tráfego de dados críticos e garantir a entrega dos dados. Por fim, é importante dar importância à segurança e monitorização da rede, ou seja, empregar medidas de segurança para evitar acesso não autorizado, violação de dados e interrupções na comunicação de rede.

UPS - Uninterruptible Power Supply

A UPS, é um aparelho elétrico que providencia um backup de energia elétrica para dispositivos ou sistemas que se encontram conectados quando a fonte de energia principal sofre uma interrupção, flutuações de tensão ou outros problemas de qualidade de energia. A UPS atua como uma ponte entre a fonte de energia principal e os dispositivos que está a proteger e abranger, garantindo assim que os mesmos recebam uma fonte de energia consistente e ininterrupta. O principal objetivo de uma UPS é evitar a interrupção ou perda de sistemas, dispositivos e dados críticos devido a uma interrupção da energia. Serve como uma rede de segurança que permite que os sistemas continuem a funcionar sem qualquer tipo de problema, mesmo quando a fonte de energia principal está comprometida. Após entender o que é uma UPS e qual o seu propósito, há que refletir que todos os sistemas têm as suas vantagens e desvantagens. Portanto, de seguida, vamos debater sobre as mesmas. (McFarlane 2022)

- Como vantagens, podemos referir:
  - Fornecimento de energia ininterrupta aos dispositivos conectados, garantindo que os sistemas críticos permaneçam operacionais durante a interrupção ou flutuação de energia.
  - Evitam a perda ou corrupção de dados que pode ocorrer quando os dispositivos são desligados repentinamente devido a interrupções de energia.
  - Garantem que os dispositivos recebam energia estável e regulada, protegendo-os de possíveis danos causados por picos de tensão, surtos ou quedas.
  - Oferecem tempo de execução suficiente para iniciar os procedimentos de *shut-down* adequados para os dispositivos conectados.
  - Ao fornecer energia condicionada e proteção contra problemas relacionados à energia, os sistemas UPS estendem a vida útil do equipamento eletrónico.
  - Nas configurações de rede e comunicação, os sistemas UPS mantêm a conectividade e evitam interrupções na transmissão de dados.
  - As empresas podem continuar a operar durante quedas de energia, reduzindo o tempo de inatividade e mantendo a produtividade.
  - Podem evitar danos ao hardware causados por perda repentina de energia, potencialmente salvando as organizações de reparos ou substituições dispendiosas.
- Por outro lado, em termos de desvantagens temos:
  - Requerem um investimento inicial, incluindo o custo do próprio dispositivo e qualquer instalação ou configuração necessária.

- Necessitam de manutenção periódica, incluindo substituições de baterias, para garantir um desempenho confiável. As substituições de baterias podem ser caras e demoradas.
- O tempo de execução fornecido por uma UPS depende da capacidade da bateria e da carga de energia. Sistemas de alta potência podem ter tempos de execução mais curtos.
- Com o tempo, as baterias da UPS degradam-se e perdem a sua capacidade, reduzindo a capacidade do sistema de fornecer energia de reserva.
- Ocupam espaço físico sendo que em sistemas maiores com tempos de execução mais longos podem ser bastante volumosas.
- Estes equipamentos, requerem o descarte ou reciclagem das baterias, contribuindo para o lixo eletrónico e preocupações ambientais.
- Embora os sistemas UPS protejam contra problemas relacionados à energia, eles não podem evitar todos os tipos de falhas, como falhas de software ou mau funcionamento de hardware.
- A instalação e configuração de sistemas UPS pode exigir conhecimento técnico, especialmente para instalações maiores ou configurações complexas.
- Podem não ser adequados para quedas de energia muito longas, pois o tempo de execução da bateria é limitado. Nesses casos, a utilização de geradores pode ser mais apropriado.

Battery Backup System with Automatic Transfer Switch

Esta solução envolve a configuração de um sistema de *backup* de bateria com uma *Automatic Transfer Switch* (ATS). A bateria de *backup* fornece energia para o sistema crítico durante interrupções, enquanto o ATS monitoriza a fonte de alimentação de entrada e muda para a fonte de energia de *backup* sem problemas quando a energia principal sofre uma queda.
Este tipo de sistema tem como funcionalidade a operação normal, deteção da queda de energia, alternância da energia principal para a bateria e monitorização e alertas. De forma mais detalhada, o sistema opera na fonte de alimentação principal e o banco de baterias é mantido carregado. Quando a fonte de alimentação principal é interrompida, o ATS deteta a queda. Seguidamente, o ATS aciona uma troca contínua para energia da bateria ativando o inversor. Por fim, o sistema monitorização e controlo acompanha os níveis de bateria, o estado da energia principal e a integridade do sistema. Ele pode enviar alertas se os níveis de bateria estiverem a ficar baixos ou se houver algum problema. Após introduzir esta solução e detalhar qual o seu propósito, é importante refletir que todos os sistemas têm as suas vantagens e desvantagens. Desta forma, seguidamente vamos debater sobre as mesmas.

- No que diz respeito a vantagens, podemos apresentar que:
  - A principal vantagem deste sistema é a capacidade de fornecer energia ininterrupta para sistemas críticos durante quedas e falhas de energia, evitando assim tempo de inatividade ou perda de dados.
  - O ATS permite uma transição suave e automática da fonte de alimentação principal para a bateria de reserva, garantindo o mínimo de interrupção nos dispositivos conectados.

  - O sistema ajuda a proteger dados confidenciais, permitindo que os dispositivos sejam desligados normalmente durante uma interrupção, minimizando o risco de corrompimento ou perda de dados.
  - O sistema de backup da bateria fornece energia por um determinado período, permitindo que os sistemas críticos continuem a operar até que a energia principal seja restaurada.
  - Ao evitar a perda abrupta de energia e reduzir o impacto das flutuações de tensão, o sistema pode prolongar a vida útil dos dispositivos conectados e reduzir a necessidade de reparos ou substituições de equipamentos.
  - Muitos sistemas modernos oferecem recursos de monitorização e alerta que notificam os administradores sobre eventos de energia, permitindo assim uma resposta e gestão pro-ativa.

- Por outra perspetiva, as desvantagens deste sistema passam por:
  - A implementação de um sistema de backup de bateria com um ATS requer um investimento inicial em equipamentos, instalação e configuração.
  - As baterias requerem manutenção e substituições periódicas para garantir um desempenho confiável. Isso pode aumentar o custo geral ao longo do tempo.
  - O tempo de execução é limitado pela capacidade das baterias. Sistemas maiores com tempos de execução mais longos podem exigir mais espaço e potencialmente ter um custo maior associado.
  - Com o tempo, as baterias degradam-se, reduzindo a sua capacidade e desempenho geral. Logo, a substituição regular das bateria são necessárias.
  - O descarte e a reciclagem de baterias representam preocupações ambientais devido aos materiais tóxicos nas baterias.
  - Embora o sistema resolva problemas relacionados à energia, ele não pode evitar todas as formas de falhas do sistema, como imprevistos de software ou mau funcionamento de hardware.
  - Instalar e configurar um sistema de backup de bateria com ATS pode exigir conhecimento técnico, especialmente para instalações maiores ou mais complexas.
  - A energia é limitada pela capacidade da bateria, tornando menos adequado para interrupções prolongadas de energia. O uso de geradores de backup podem ser necessários para suporte de energia mais duradouro.
  - A eficácia do sistema depende da confiabilidade do ATS, inversor, baterias e outros componentes. Uma falha em qualquer um desses componentes pode comprometer o desempenho do sistema.

Redundant Power Sources with Dual Power Supplies

Neste tipo de solução, o sistema em causa é equipado com fontes de alimentação redundantes conectadas às fontes de alimentação separadas. Se uma fonte de energia falhar, o sistema automaticamente consegue alternar para a fonte de backup, garantindo assim uma operação ininterrupta.
A funcionalidade desta solução, é que a mesma opera utilizando a energia de alimentação

principal. Caso a fonte de alimentação principal falhe, o mecanismo de comutação automática ativa a fonte de alimentação de reserva sendo que, o sistema alterna perfeitamente para a fonte de alimentação de backup, evitando assim o tempo de inatividade. O sistema continua a operar utilizando a fonte de alimentação de backup até que a alimentação principal seja restaurada. Por fim, o sistema de monitorização rastreia o estado da fonte de alimentação e a integridade do sistema, sendo capaz de enviar alertas caso seja necessário. Após a apresentação desta solução e quais as suas principais funcionalidades, é necessário refletir as vantagens e desvantagens.

- Como pontos positivos podemos referir que:
  - Esta solução oferece confiabilidade, ou seja, com fontes de alimentação redundantes e fontes de alimentação duplas, o sistema permanece operacional mesmo se uma fonte de alimentação ou fornecimento falhar.
  - Fontes de energia redundantes garantem que os sistemas críticos tenham uma fonte de alimentação ininterrupta, minimizando o tempo de inatividade e as interrupções.
  - Se uma fonte de alimentação ou fonte apresentar problemas, o sistema muda automaticamente para a fonte de backup, garantindo uma operação contínua sem intervenção manual.
  - O sistema torna-se mais resistente a falhas na fonte de alimentação, garantindo a estabilidade e o desempenho dos dispositivos conectados.
  - Os sistemas podem ser projetados com vários níveis de redundância, permitindo que as organizações dimensionem a redundância para atender aos seus requisitos e orçamento.
  - Fontes de alimentação duplas e fontes redundantes permitem mecanismos de *failover* controlados, minimizando o impacto nas operações.
  - Durante a manutenção ou substituição de uma fonte de alimentação, a outra fonte pode continuar a alimentar o sistema, reduzindo a necessidade de tempo de inatividade programado.
- Numa perspetiva de pontos negativos, é importante ter em consideração que:
  - A implementação de fontes de alimentação redundantes e fontes de alimentação duplas acarreta custos iniciais mais altos para hardware e infraestrutura adicionais.
  - Gerir e configurar fontes de alimentação redundantes, fontes de alimentação duplas e mecanismos de *failover* associados pode ser complexo e exigir conhecimento especializado.
  - A gestão e a manutenção de várias fontes e aprovisionamentos de energia podem ser mais confuso e o diagnóstico de problemas pode ser mais complexo.
  - Manter duas fontes de energia e suprimentos pode levar a um maior consumo de energia e custos operacionais.
  - A eficácia do sistema de alimentação redundante depende da confiabilidade das fontes de alimentação e mecanismos de *failover*. Uma falha em qualquer um desses componentes pode afetar todo o sistema.

- Projetar e implementar a lógica de *failover* requer consideração cuidadosa para garantir o acionamento adequado de *failover* e recuperação oportuna.
- Erros humanos, como configurações incorretas ou configurações de *failover* incorretas, podem afetar a eficiência do sistema de energia redundante.

3.3.3 Análise critica

Após a introdução das falhas a que os sistemas se encontram suscetíveis e a identificação das mesmas, foi possível obter abordagens e possíveis soluções para os diferentes tipos de falhas.
Depois de identificar essas possíveis soluções foi necessário realizar um estudo, tendo em consideração as funcionalidades, vantagens e desvantagens de cada abordagem, para ser possível alcançar a conclusão de qual destas abordagens pode resultar numa melhor performance num sistema como o que foi desenvolvido no projeto, não desprezando tópicos como a complexidade, instalação, custo, entre outros.

Abordagens Propostas	Complexidade	Instalação	Manutenção	Custos	Resposta à falha
Uninterruptible Power Supply	Média	Média	Média	Alto	Rápida
Battery Backup System with ATS	Alta	Média	Média	Alto	Mediano
Redundant Power Sources with DPS	Alta	Média	Média	Alto	Mediano

Tabela 3.1: Comparação das abordagens propostas

Após a comparação que está presente na tabela 3.1, foi possível identificar qual destas abordagens propostas é a mais indicada. Quando comparamos os campos da tabela e as vantagens e desvantagens associadas a cada abordagem, podemos concluir que, para este projeto, o uso de um sistema como o UPS seria o mais indicado.

Esta conclusão foi retirada devido ao facto de as outras duas abordagens apresentarem uma probabilidade maior de falhas devido às configurações necessárias sendo que, se essas configurações não forem bem pensadas e executadas a performance pode estar em causa. No que se refere a custos, todas as abordagens têm um grau elevado devido às baterias, uma vez que, quanto melhor forem as baterias, maior capacidade e eficácia o sistema vai apresentar. Ao nível da manutenção, é muito importante ter em atenção devido ao facto de que é fundamental saber em que estado se encontram as baterias para, no caso de ser necessário, seguir para a sua substituição. Uma bateria defeituosa ou avariada prejudica o bom funcionamento do sistema.

3.3.4 Arquitetura

Relativamente à possível arquitetura no caso da implementação do sistema UPS, passaria por algo idêntico ao que se encontra presente nas figuras seguintes.

A figura 3.2, representa uma situação normal de uso, ou seja, a fonte de alimentação principal está funcional e disponível para o sistema, permitindo o bom funcionamento do mesmo e em segundo plano, carregar as baterias de backup e fontes de alimentação secundárias caso seja necessário as mesmas estarem na sua capacidade total.

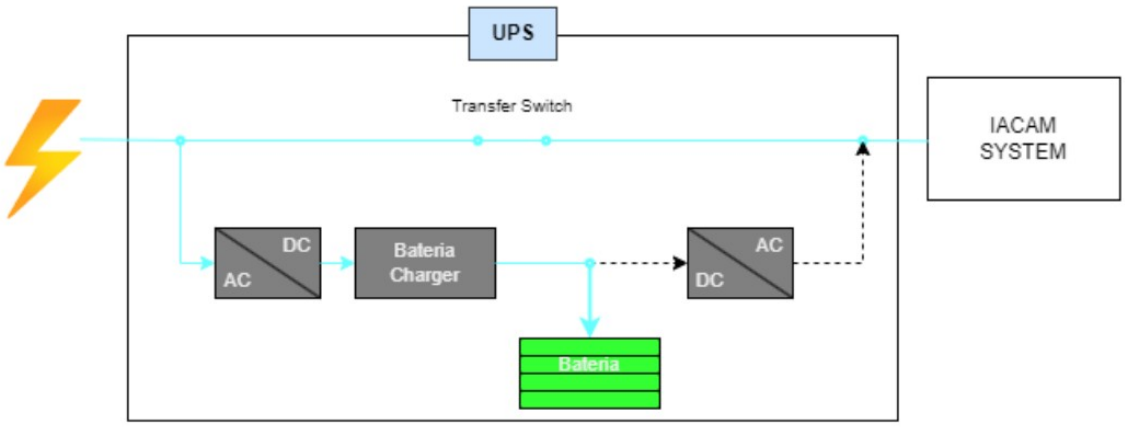


Figura 3.2: UPS desativo em caso de falha

Enquanto que a figura 3.3 representa uma situação de falha e de como o sistema UPS responde perante esta situação. As baterias, que num momento normal estão a ser carregadas em segundo plano, deixam de ser uma fonte de alimentação secundária e de backup para uma fonte de alimentação principal.

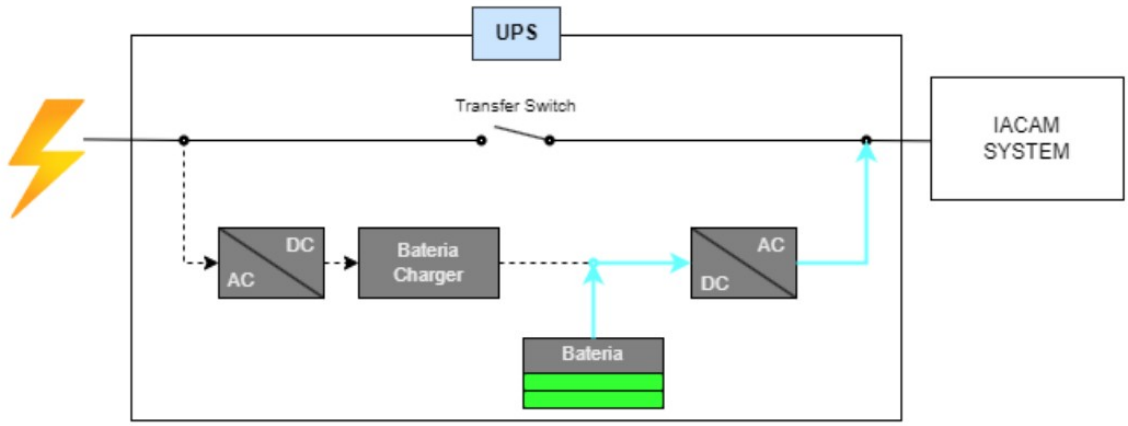


Figura 3.3: UPS ativo em caso de falha

Capítulo 4

Desenvolvimento

Este capítulo tem como principal objetivo a descrição de todo o processo de desenvolvimento do projeto, desde qual foi o trabalho realizado ao longo do semestre, que tipos de problemas foram surgindo durante este processo e quais foram as soluções encontradas e que alterações surgiram, desde a sua arquitetura, lógica entre outros, quando comparado com o plano inicial.

4.1 Trabalho realizado

Relativamente ao trabalho desenvolvido no decorrer do projeto, o mesmo está divido em diferentes categorias bem como, o *firmware* que está presente no dispositivo IoT, a aplicação web que tem como objetivo tornar mais amigável a interpretação dos dados e a interação com alguns componentes do sistema, a aplicação FIWARE que se encontra presente no servidor principal e, por fim, a parte de *scripting* que simula a obtenção de dados dos servidores que estão presentes nos bastidores.

4.1.1 Firmware - Dispositivo Iot

Concepção do Firmware

O desenvolvimento de um *firmware* tem o principal propósito de possibilitar que o micro controlador, neste caso o ESP32, seja capaz de obter dados dos sensores, processá-los, receber dados externos desde APIs de terceiros bem como dados referentes dos servidores, tomar uma decisão e, por fim, enviar os dados para o *Context Broker* da aplicação FIWARE para que esses dados sejam consumidos pela aplicação web.

Sensor DHT22

Relativamente a esta parte do código, existem três etapas para que a obtenção de dados relativos à temperatura ambiente e humidade relativa sejam possíveis. A primeira parte, que se encontra implementada no *init_setup*, foi denominada de *SetupDHT()* e tem como propósito inicializar o próprio sensor dht22. Após isso no *init_loop* é chamada a função *dht()* que tem o objetivo de obter os valores da temperatura e humidade. De seguida, ainda dentro da mesma, é chamada a função *ErrorDHT()* em que o seu foco é verificar se existiram erros no processo de leitura e obtenção de dados.

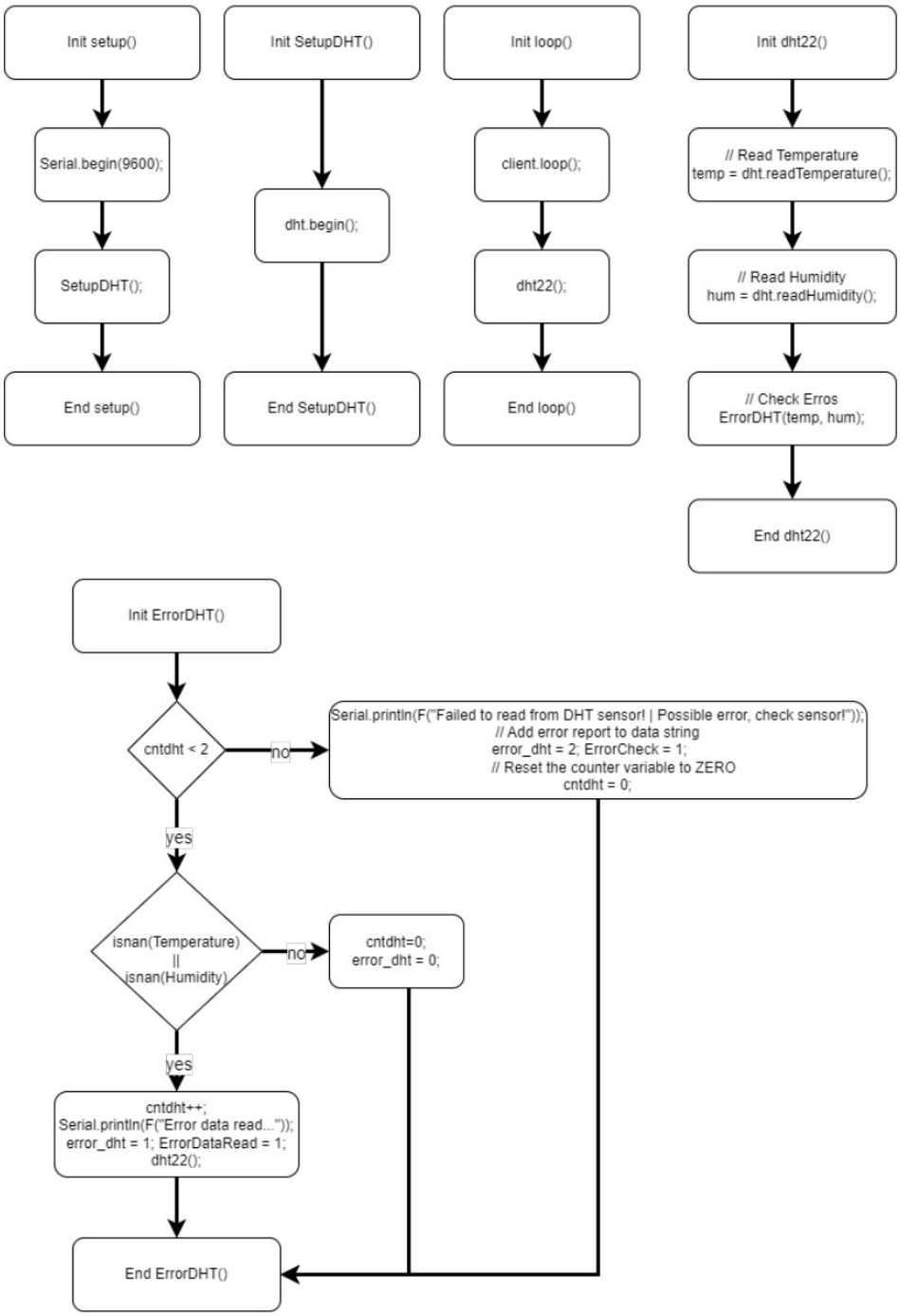


Figura 4.1: Fluxogramas do DHT22

Sensor CO

Nesta secção do *firmware*, o objetivo é de detetar o monóxido de carbono dentro da sala dos servidores. Esta parte do código está dividido em duas etapas, uma dessas etapas é a calibração do sensor que tem o objetivo de calibrar o sensor para que a obtenção dos dados seja mais precisa e a outra etapa é a obtenção da concentração do monóxido de carbono (CO).

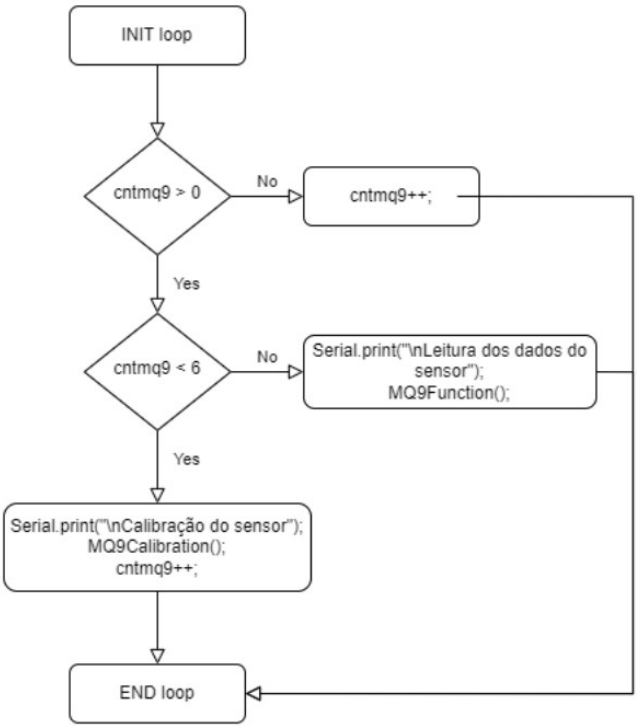


Figura 4.2: Função Loop

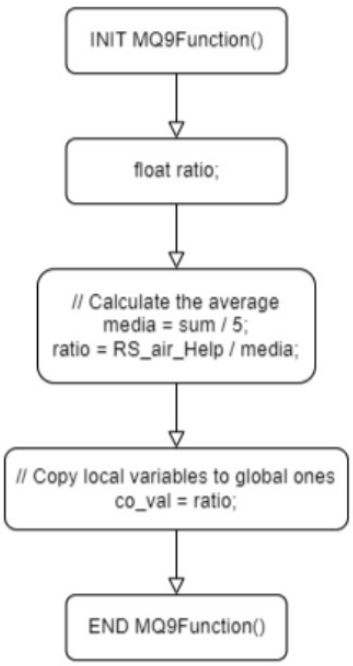


Figura 4.3: Função de obtenção de dados

MQTT

Nesta secção do *firmware*, existe a componente da subscrição de tópicos e a componente de publicação de tópicos.

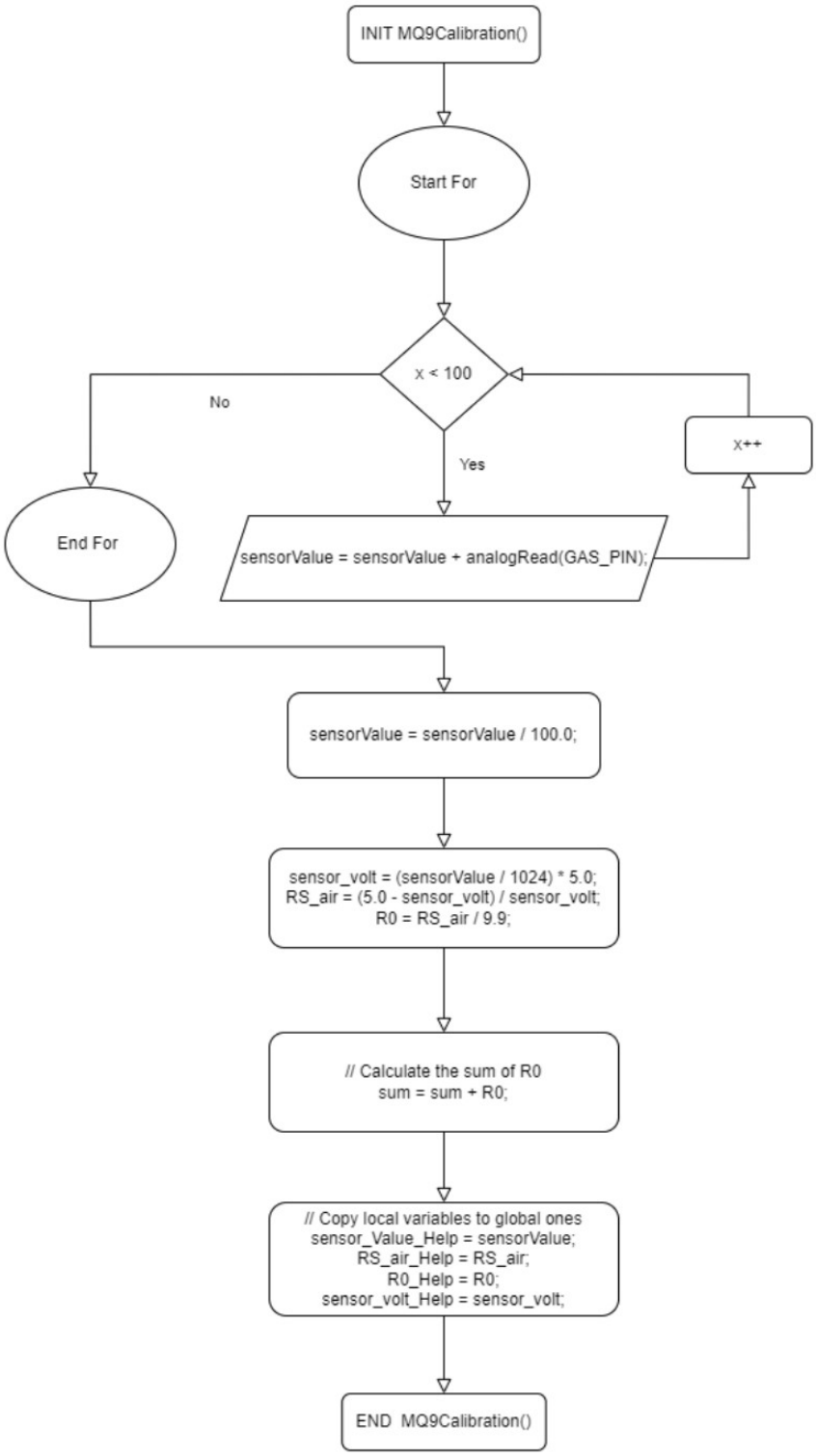


Figura 4.4: Função de calibração do sensor

Na componente da subscrição de tópicos, a mesma tem o seu foco na obtenção de dados exteriores para o enriquecimento da lógica presente na secção de tomada de decisão. Enquanto a componente de publicação de tópicos tem por princípio enviar os dados recolhidos dos sensores para os tópicos em questão.

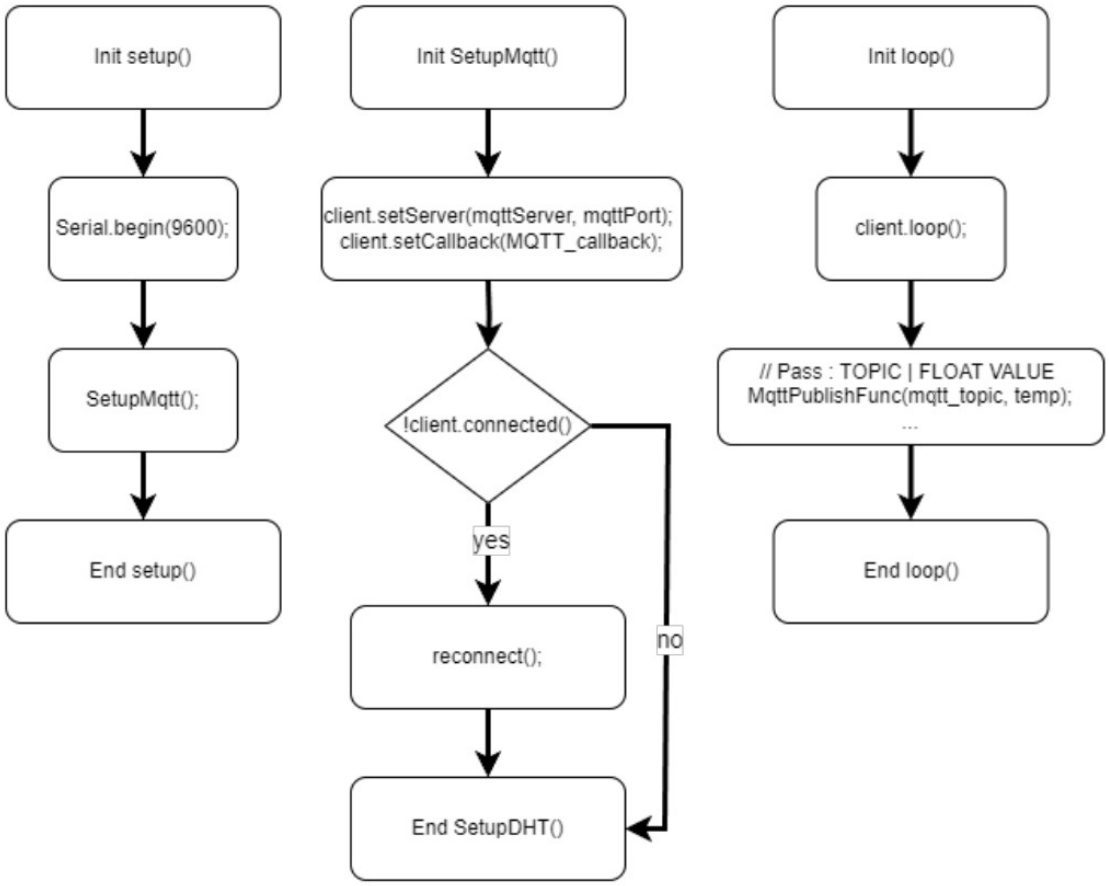


Figura 4.5: Fluxogramas do Mqtt

Interrupts

Neste micro controlador é possível definir um função de rotina de serviço de interrupção (ISR), que eventualmente vai ser chamada quando o pino GPIO mudar o seu nível lógico. Todos os pinos GPIO na placa ESP32 podem ser configurados para atuar como entradas de solicitação de interrupção.

Para a realização dos *interrupts*, foi utilizado a função **attachInterrupt()** para definir uma interrupção pino por pino. A sintaxe da função é "**attachInterrupt(GPIO_Pin, ISR, Mode)**"e como é possível visualizar, a função aceita três argumentos:

- GPIO_Pin:
  - Define o pino GPIO como o pino do *interrupt*.
- ISR:
  - Diz respeito ao nome da função que será chamada toda a vez que o *interrupt* ocorrer.
- Mode:
  - Define quando o *interrupt* deve ser acionado. Sendo que existem constantes que são predefinidas como valores válidos.

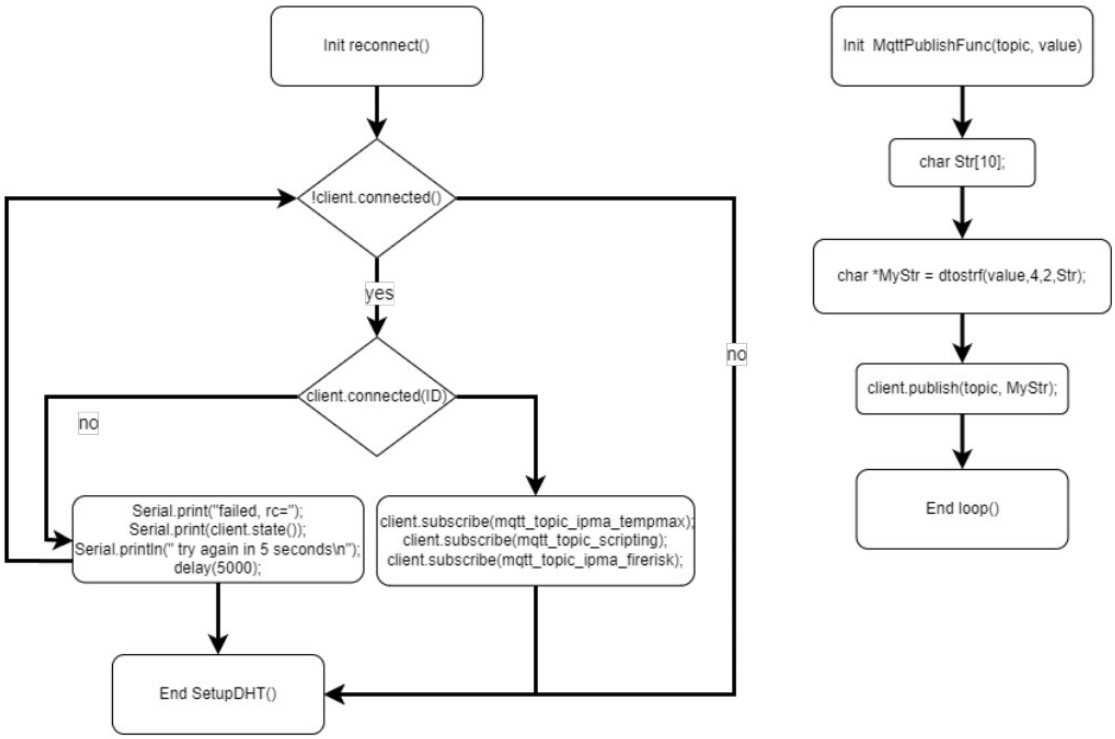


Figura 4.6: Fluxogramas do Mqtt

* *LOW* - Aciona o *interrupt* sempre que o pino é *LOW*.
* *HIGH* - Aciona o *interrupt* sempre que o pino é *HIGH*.
* *CHANGE* - Aciona o *interrupt* sempre que o pino muda de valor, de *HIGH* para *LOW* ou *LOW* para *HIGH*.
* *FALLING* - Aciona a interrupção quando o pino vai de *HIGH* para *LOW*.
* *RISING* - Aciona a interrupção quando o pino vai de *LOW* para *HIGH*.

Após esta breve introdução do que são os *interrupts*, no *firmware* desenvolvido foi necessário implementar esta função com o intuito de quando o sensor de chama detetar uma fonte de calor aciona um *interrupt* com o modo *RISING* pois o sensor passa do valor lógico zero para o valor lógico um. E quando o sensor deixa de detetar uma fonte de calor também aciona um *interrupt* com o modo *FALLING* pois o sensor passa do valor lógico um para o valor lógico zero. (LastMinuteEngineers 2022)

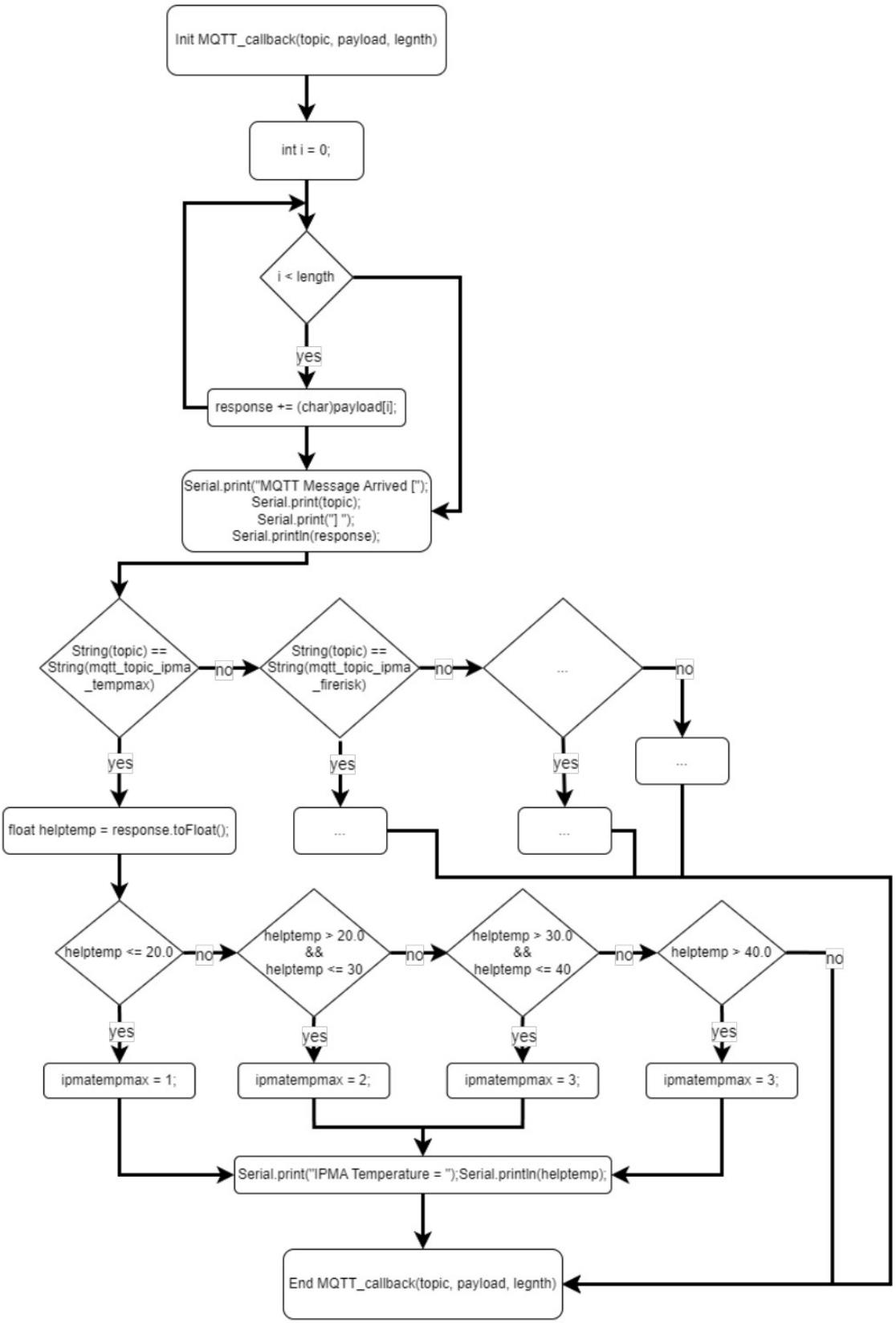


Figura 4.7: Fluxogramas do Mqtt

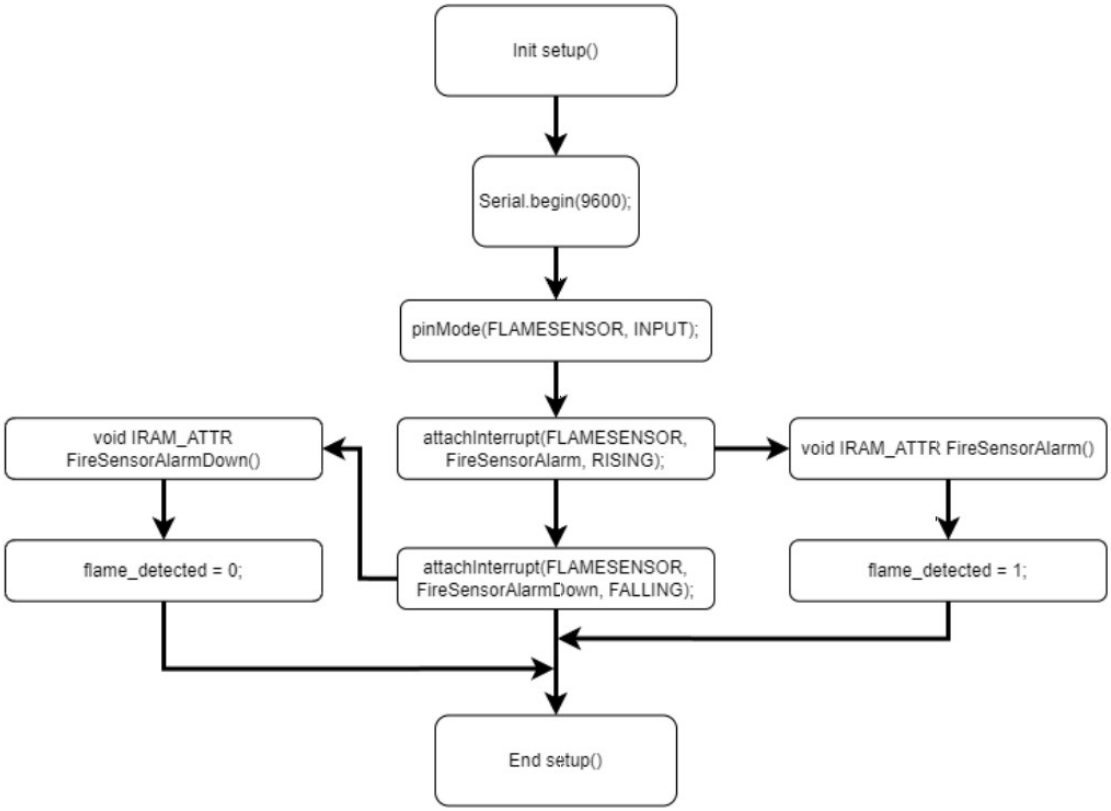


Figura 4.8: Sensor de deteção de chama

Tomada de Decisão

Nesta secção, o intuito é que o dispositivo IoT face as condições, recolha de dados e análise dos mesmos, seja capaz de tomar uma decisão de forma autónoma.
Para isso ser possível, inicialmente é avaliado em que parâmetro a média dos valores de temperatura dos processadores está e atribuir-lhe um valor. Após isso, é feito o mesmo processo desta vez com o valor da temperatura ambiente recolhida com recurso ao sensor DHT22. Também é feito um cálculo relativamente à informação sobre o risco de incêndio e a temperatura máxima, dados obtidos com a ajuda da API do IPMA.
Depois de obter todos esses valor atribuídos, é chamada a função *CalcProb()* com o intuito de calcular um valor índice que depois será usado na definição da nova leitura que o ESP32 faz de seguida, ou seja, esse valor, denominado *TimeSet*, tem diferentes tempos associados a situações mais críticas ou menos críticas.

- Valores possíveis do TimeSet:
  - **DEFAULT** - Quarenta e cinco minutos
  - **BAIXO** - Trinta e cinco minutos
  - **MÉDIO** - Vinte minutos
  - **ALTO** - cinco minutos

– **CRÍTICO** - Um minuto

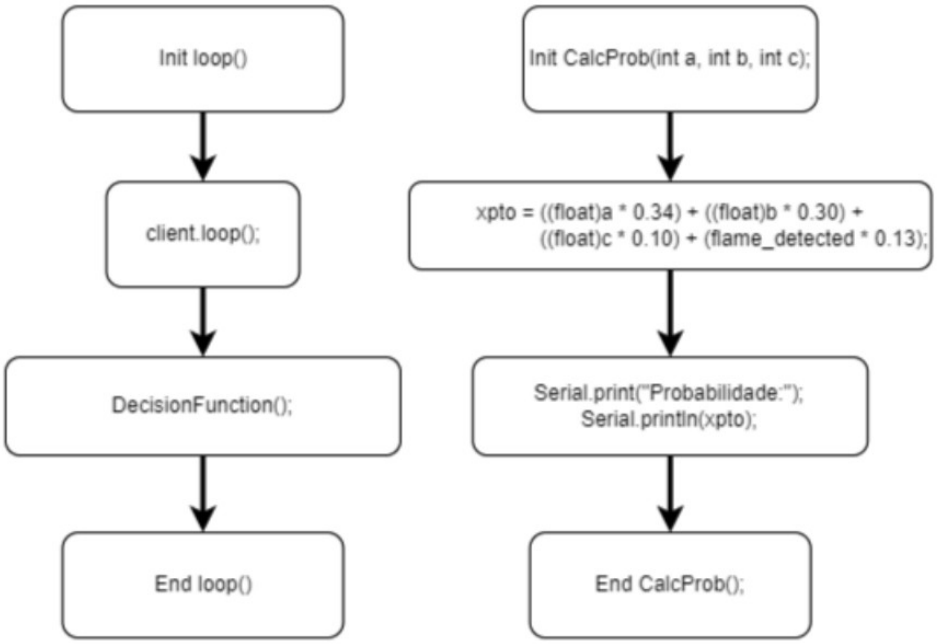


Figura 4.9: Fluxogramas referentes à tomada de decisão

Constituição do dispositivo IoT

Relativamente à constituição do dispositivo, foram necessários alguns componentes que por si só apresentam a sua precisão e desvio padrão. Estes fatores foram considerados no desenvolvimento do projeto e na tomada de decisão.

De seguida, todos os componentes vão ser identificados onde é detalhada a sua função no sistema e as suas características.

Sensor DHT22:

- Função:
  - O Sensor DHT22 é utilizado para medir a temperatura ambiente numa escala entre os -40º a +80º graus Celsius e a humidade do ar nas faixas de 0 a 100%, contando com uma precisão que varia de 2 a 5%.
- Características:
  - Tensão da operação: 3-5VDC (5,5VDC máximo),
  - Faixa de medição de humidade: 0 a 100% UR,
  - Faixa de medição de temperatura: -40º a +80ºC,
  - Corrente: 2,5mA max durante uso, em stand by de 100uA a 150 uA,
  - Precisão de medição da humidade : ± 2,0% UR e

  - Precisão de medição da temperatura: ± 0,5 ºC.

Sensor de gás MQ-9 - Monóxido De Carbono:

- Função:
  - MQ-9 é um sensor de baixo custo e de alta sensibilidade para monóxido de carbono e gases combustíveis (metano e LPG). É apropriado e tem como foco a deteção de monóxido de carbono e fazer a sua monitorização. O monóxido de carbono é detetado dentro da faixa 10 a 1000 ppm de CO.
- Características:
  - Tensão loop Vc: <= 10V DC,
  - Tensão de aquecimento Vh: 5.0V ± 0.2V ACorDC,
  - Resistência do aquecedor: 31 ohm ± 3 ohm (temperatura ambiente) e
  - Aquecedor de consumo de energia PH: <= 900mW.

Sensor de Chama OKY3053:

- Função:
  - Este sensor é sensível à chama. É geralmente usado como detetor de chamas, sendo esse o seu motivo para ser um dos componentes deste projeto.
- Características:
  - Deteta uma chama com um comprimento de onda na faixa de 760nm-1100nm,
  - Ângulo de deteção de cerca de 60 graus, é sensível ao espectro de chama,
  - Precisão ajustável e
  - Tensão de operação 3.3V ~5V.

Relé - Sinal 5V 2 Canais 220V/10A:

- Função:
  - Este Módulo Relé 5V com 2 canais é possível controlar diversos dispositivos de corrente alternada, de até 10 A. O módulo é equipado com transístores, conectores, LEDs, díodos e relés de alta qualidade. Cada canal possui um LED para indicar o estado da saída do relé.
- Características:
  - Tensão da operação: 5VDC,
  - Permite controlar cargas de 220V AC,
  - Corrente típica da operação: 15 20mA,
  - LED indicador de status,

  - Tensão de saída: (30 VDC a 10A) ou (250VAC a 10A) e
  - Tempo de resposta: 5 10ms.

LED 5mm:

- Função:
  - Este componente tem o propósito de simular a falta do mecanismo de uma janela mecânica. Que tem o objetivo de ajudar no arrefecimento do espaço onde se encontram os servidores. Apresenta apenas dois estados, 0 (zero) ou desligado e 1 (um) ou ligado.
- Características:
  - Tipo de díodo LED,
  - Diâmetro do díodo LED 5mm,
  - Clareza 3800-5500mcd,
  - Tensão de trabalho 2.8...4V,
  - Corrente do díodo LED 20mA e
  - Ângulo de iluminação 15°.

Ventoinha:

- Função:
  - Este componente também apresenta o propósito de ajudar no arrefecimento do espaço onde se encontram os servidores. Bem como o componente anterior, este também apresenta apenas dois estados, 0 (zero) ou desligado e 1 (um) ou ligado.
- Características:
  - Tensão de funcionamento: 12Vdc,
  - Fluxo do ar: 12m^3/hr,
  - Ruído: 25dBA,
  - Corrente: 0.1A e
  - Rotação: 6000 rpm.

Protótipo do dispositivo

Este segmento tem o foco de apresentar o protótipo que foi desenvolvido ao longo deste projeto. Devemos ter em consideração que esse mesmo protótipo não passa de algo muito primórdio, uma vez que esse não foi o foco do projeto. Desta forma, a realização e criação de um estudo com foco no design encaixa no tópico do trabalho futuro.

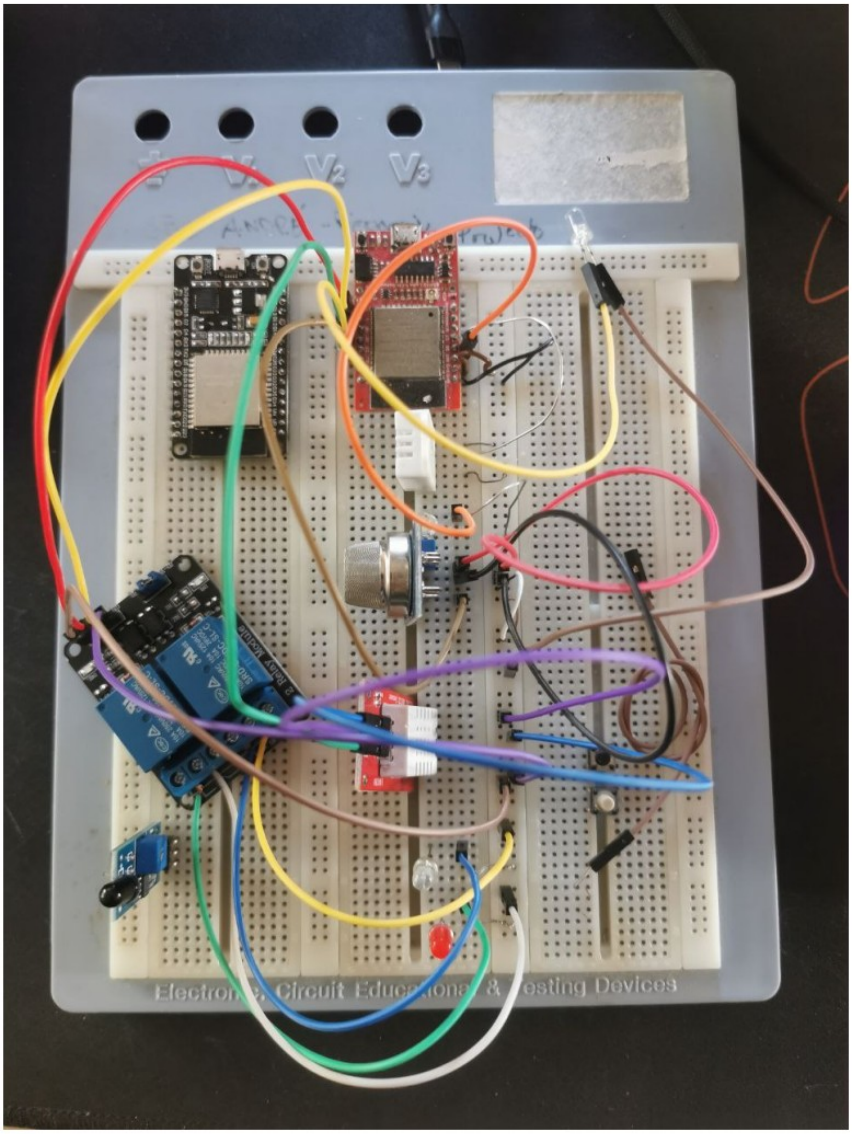


Figura 4.10: Protótipo

Output do Firmware

Esta secção tem o propósito de exibir o *output* fornecido pelo *firmware* para tornar a interpretação da lógica e da obtenção de dados do dispositivo IoT mais clara e intuitiva.

Inicialmente, o dispositivo necessita de se conectar à rede WIFI e ao *Context Broker* MQTT. Estando esse mesmo passo representado na imagem 4.11.

```
17:08:50.231 -> Connecting to Bitas
17:08:50.800 -> ......
17:08:53.338 -> WiFi connected
17:08:53.338 -> IP address:
17:08:53.385 -> 192.168.1.77
17:08:53.385 ->
17:08:53.385 -> Connect to host 192.168.1.128 in port 1883
17:08:53.426 -> Attempting MQTT connection...
```

Figura 4.11: Conexão à rede e mqtt

De seguida, começam a ser exibidos os dados referentes à recolha dos diversos valores obtidos pelos sensores.
Como é possível visualizar na imagem que se segue, que a mesma exibe os valores da temperatura ambiente e da humidade relativa obtidas através do sensor DHT22.

```
17:11:33.806 -> ################################################################################
17:11:33.915 -> ##############################################  INIT LOOP  ##################
17:11:34.059 -> ################################################################################
17:11:34.159 ->
17:11:34.159 -> ################################################################################
17:11:34.292 -> ############################################## IACAM001 ##################
17:11:34.392 -> ################################################################################
17:11:34.525 ->
17:11:34.525 -> Sensor DTH22 Values:
17:11:34.525 -> ________________________________________________________________________________
17:11:34.659 ->
17:11:34.659 ->         -> Temperature: 25.10 Celsius
17:11:34.692 -> ________________________________________________________________________________
17:11:34.793 ->
17:11:34.793 ->         -> Humidity: 77.30 %
17:11:34.826 -> ________________________________________________________________________________
17:11:34.925 ->
17:11:34.925 ->
17:11:34.925 -> ################################################################################
17:11:35.059 -> ################################################################################
17:11:35.159 -> ################################################################################
```

Figura 4.12: Obtenção de dados dht22

No output, também é visível se durante o processo de obtenção de dados o sensor dht22 apresentou alguma falha ou se o mesmo está funcional. Neste último caso, o sensor não apresenta nenhum tipo de erro.

```
17:11:37.625 -> ################################################################################
17:11:37.725 -> ################################################################################
17:11:37.859 -> ################################################################################
17:11:37.959 ->
17:11:37.959 ->  Erros no Sistema :
17:11:37.992 -> ________________________________________________________________________________
17:11:38.092 ->
17:11:38.092 ->         -> Mensagens de erros no DHT22:
17:11:38.125 ->  * Não há erros no sensor :)
17:11:38.159 ->  *
17:11:38.159 ->  *
17:11:38.192 -> ________________________________________________________________________________
17:11:38.292 ->
```

Figura 4.13: Erros dht22

A imagem 4.14, representa o momento em que é detetada alguma fonte de calor

ou chama no espaço onde se encontra o sensor. Neste caso específico, nenhuma fonte de calor ou chama foi detetada.

Figura 4.14: Deteção de chamas

Caso ocorra algum incidente ou exista a possibilidade de vir a ocorrer, é necessário alertar os administradores. Para isso ser possível, é necessário enviar essa informação para o servidor, com recurso a funções, e posteriormente alertar os próprios.
Esse processo está representado na imagem 4.15.

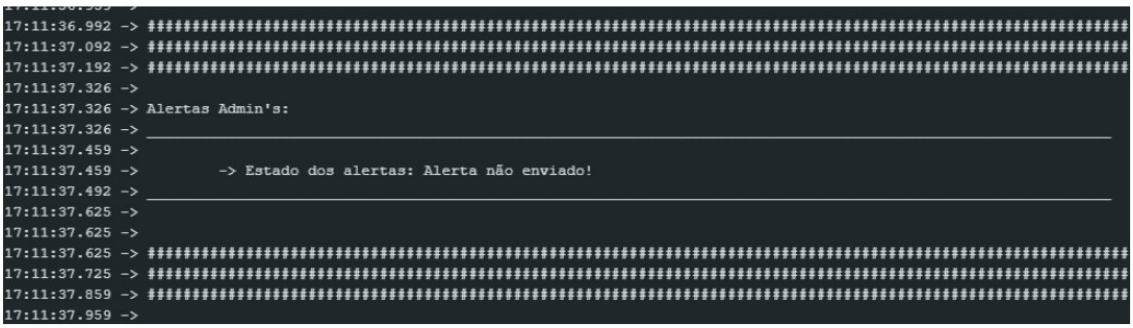


Figura 4.15: Alerta ao administrador

Também é visível o estado dos atuadores, ou seja, é exibido se o atuador está ligado ou desligado. Na figura 4.17 está presente essa metodologia.

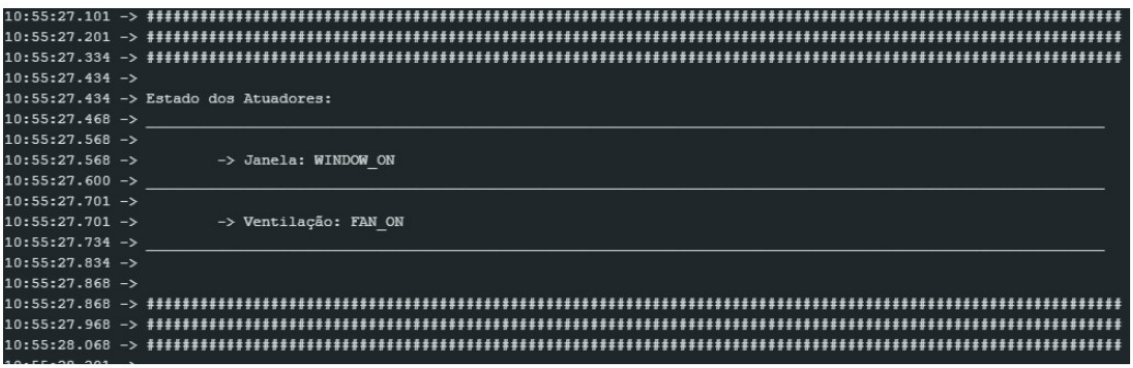


Figura 4.16: Estado dos atuadores

Por fim, também é possível observar os dados obtidos pelo sensor MQ-9, que tem como principal objetivo a obtenção da concentração de monóxido de carbono.

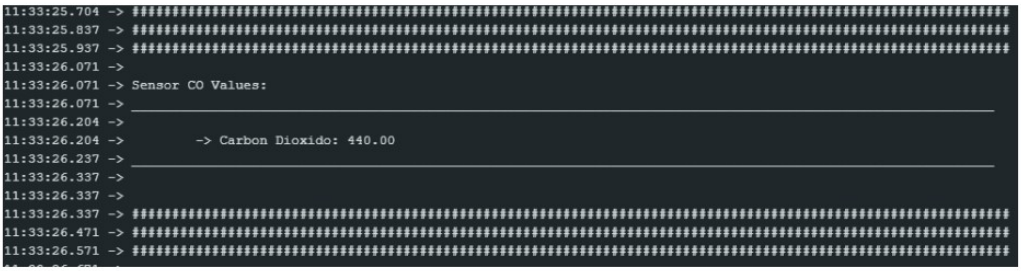


Figura 4.17: Estado dos atuadores

4.1.2 Aplicação Web

No que diz respeito à aplicação web, a mesma foi desenvolvida com a *framework* Angular. Esta aplicação tem dois componentes, sendo que o componente principal é o **dashboard** cujo objetivo é tornar mais percetível e amigável a interpretação dos dados de todo o sistema. Referente ao segundo componente desta aplicação, o **Actuators - Manual Control** foca-se na exibição do status dos atuadores presentes no dispositivo IoT e dando a possibilidade ao utilizador de ligar ou desligar de forma manual. Contudo, de seguida, cada componente será detalhado individualmente.

Componente - Dashboard

Neste componente é possível observar dados da temperatura, máxima e mínima, da probabilidade de precipitação e o risco de incêndio para o dia atual. Estes dados são obtidos através de um pedido GET à API do IPMA em que o output é retornado no formato *json*. Após fazer o pedido GET e guardar o *payload* é invocada uma função, **TestFunc**, que tem o propósito de manipular os dados recebidos e associar esses mesmos dados a cada campo de texto no *dashboard*.

```
1       const apiUrl = 'http://api.ipma.pt/open-data/forecast/meteorology/
    cities/daily/1131200.json';
2       axios.get(apiUrl)
3       .then(response => {
4         this.TestFunc(payload.dataUpdate,payload.data[0].tMax,payload.
    data[0].tMin,payload.data[0].precipitaProb);
5       })
6       .catch(error => {
7         alert('Error code : ' + error);
8       });
9
10(some other code)
11
12  tempmax: string = 'Default';
13  tempmin: string = 'Default';
14  precprob: string = 'Default';
15  lastupd: string = 'Default';
16  TestFunc(LastUpd,TempMax,TempMin,PrecProb): void{
17    this.lastupd=LastUpd;
18    this.tempmax=TempMax;
```

```
19      this.tempmin=TempMin;
20      this.precprob=PrecProb;
21    }
```

Também é possível visualizar dados referentes ao espaço onde os servidores presentes nos bastidores estão alocados. Ou seja, dados como a temperatura ambiente da sala, a humidade relativa presente no ar, o dióxido de carbono em partes por milhão (ppm) e, por fim, se foi detetado um incêndio ou fonte de calor. Também é visível o estado atual dos atuadores presentes no sistema, ou seja, o utilizador consegue facilmente identificar que atuadores estão ligados ou desligados. Estes mesmos dados são obtidos através da aplicação *fiware* com recurso a pedidos *GET*, sendo que a lógica do preenchimentos dos campos de texto é idêntica à mencionada anteriormente.

De seguida é apresentada uma imagem relativa ao *front-end* da componente **dashboard**.

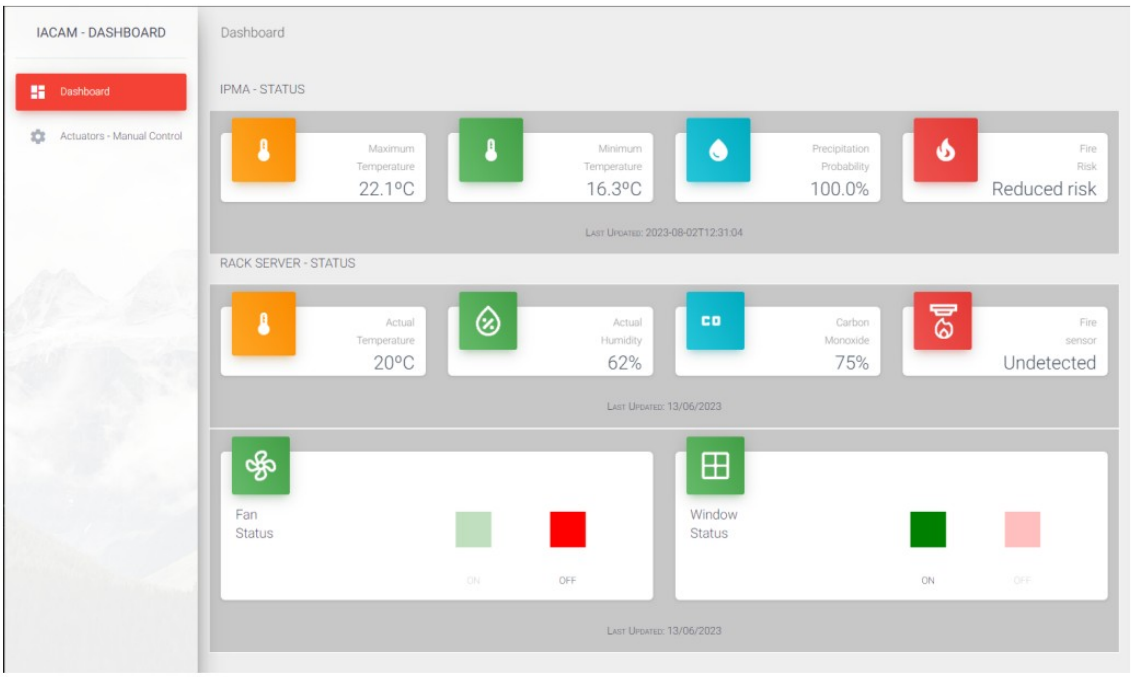


Figura 4.18: Componente *dashboard*

Componente - Actuators - Manual Control

Referente à componente **Actuators - Manual Control**, a mesma tem o objetivo de exibir, em forma de tabela, os atuadores presentes em cada dispositivo, neste caso, *Device 001*. Cada linha apresenta 4 colunas sendo elas: o **id**, o **actuator name**, o **status** e o **action**.

- **id**
  - Refere-se ao identificador (id) do atuador.
- **actuator name**
  - Refere-se ao nome do atuador.
- **status**
  - Refere-se ao estado do atuador, ligado ou desligado.
- **action**.
  - Refere-se às ações possíveis, ligar o atuador ou desligar o atuador.

Estes mesmos dados são obtidos através da aplicação *fiware* com recurso a pedidos *GET* e *POST*, sendo que a lógica do preenchimentos dos campos de texto é idêntica à mencionada anteriormente.

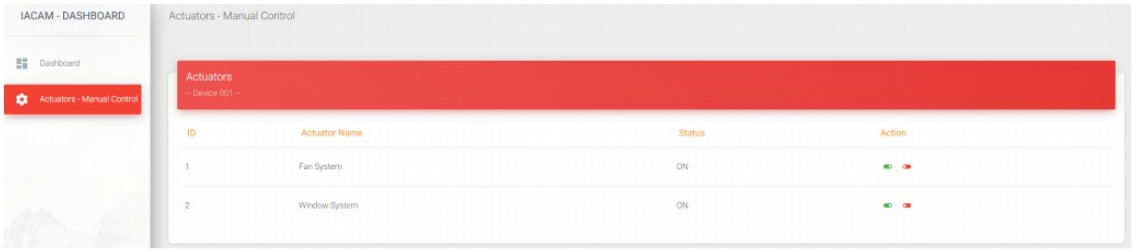


Figura 4.19: Componente Actuators - Manual Control

4.1.3 Aplicação FIWARE

Relativamente à aplicação FIWARE esta é inicializada e, posteriormente, fica em modo *running* num *container* com o suporte da plataforma Docker.

Para isso, a aplicação necessita de alguns componentes que vão ser identificados de seguida e detalhados individualmente.

Ficheiro docker-compose.yml

Neste ficheiro estão presentes as configurações dos serviços utilizados pela aplicação FIWARE, ou seja, todas as informações de configuração necessárias para conectar o *broker* Mosquitto MQTT, os dispositivos IoT e o IoT Agent podem ser vistas na secção de serviços do arquivo *docker-compose.yml*.

Aqui está um exemplo da configuração, neste caso, do IoT Agent.

```
iot-agent:
    image: fiware/iotagent-ul:latest
    hostname: iot-agent
    container_name: fiware-iot-agent
    depends_on:
      - mongo-db
    networks:
        - default
    expose:
        - "4041"
        - "7896"
    ports:
        - "4041:4041"
        - "7896:7896"
    environment:
        - "IOTA_CB_HOST=orion"
        - "IOTA_CB_PORT=1026"
        - "IOTA_NORTH_PORT=4041"
        - "IOTA_REGISTRY_TYPE=mongodb"
        - "IOTA_LOG_LEVEL=DEBUG"
        - "IOTA_TIMESTAMP=true"
        - "IOTA_MONGO_HOST=mongo-db"
        - "IOTA_MONGO_PORT=27017"
        - "IOTA_MONGO_DB=iotagentul"
        - "IOTA_PROVIDER_URL=http://iot-agent:4041"
        - "IOTA_MQTT_HOST=mosquitto"
        - "IOTA_MQTT_PORT=1883"
```

Ficheiro Dockerfile

Este ficheiro tem apenas a função de executar um comando para tornar possível que o *script* em Python fique em execução.

```
FROM python:3.10
ADD . /code
COPY . /code
WORKDIR /code
RUN pip install -r requirements.txt
CMD [ "python", "-u", "app.py"]
```

Figura 4.20: Dockerfile

4.1.4 Scripting

Esta parte do desenvolvimento tem o objetivo de simular a obtenção de dados referentes aos servidores que se encontram presentes nos bastidores. Como não foi possível a utilização de um espaço desses, foi utilizado um computador fixo antigo que tem como sistema operativo Ubuntu 16.04.7 LTS. A extração dos dados a partir daí.

Desenvolvimento do Script

O *script*, desenvolvido com a linguagem Python, inicialmente faz o *import* de algumas bibliotecas. De seguida, executa comandos para captar o output do comando *"sensors"* e escrevê-lo para um ficheiro denominado por "sensors_output.txt".

```
result = subprocess.run(["sensors"], stdout=subprocess.PIPE)
output = result.stdout.decode("utf-8")

with open("sensors_output.txt", "w") as file:
    file.write(output)
```

De seguida abrimos novamente o ficheiro de texto e é lido o conteúdo do mesmo para uma variável. Dado isso, prossegue-se para a extração dos valores da temperatura relativos a cada CPU, guardando esses valores em variáveis.

```
file = open("sensors_output.txt", "r")
content = file.read()
file.close()

temp1_start = content.find("temp1:") + len("temp1:")
temp1_end = content.find("C", temp1_start)
temp1 = content[temp1_start:temp1_end]

temp2_start = content.find("temp2:") + len("temp2:")
temp2_end = content.find("C", temp2_start)
temp2 = content[temp2_start:temp2_end]
```

Findada essa parte, é necessário preparar a *string* de dados para enviar por MQTT para o *broker* Mosquitto.

```
temp1test = temp1 + ";" + temp2
special_characters = ['+',' ']
normal_string = temp1test
for i in special_characters:
    normal_string = normal_string.replace(i,"")

broker = "192.168.1.154"
port = 1883
topic = "scripting/tempcpu

def on_publish(client, userdata, result):
    print("\nData Published!\n")
    pass
```

```
client = paho.Client("server_1")
client.on_publish
client.connect(broker,port)
ret = client.publish(topic, normal_string)
```

Output do Script

Nesta subsecção, o principal objetivo é apresentar qual o *output* devolvido quando o *script* de obtenção da temperatura dos CPU's é executado.
Logo, a figura 4.21 representa esse mesmo *output*. Como é visível na figura anterior, primeiramente é obtido os valores da temperatura dos CPU's e de seguida é preparada uma *string* que será enviada para o servidor principal com recurso ao protocolo MQTT.

```
ubuntuserver@ubuntusevrer:~/iacam$ sudo python3 Test.py
temp1:          +32.0 °C
temp2:          +53.5 °C
Message to sended: 32.0;53.5


_____________________________________________________________

Send data to topic scripting/tempcpu

MQTT value message : 32.0;53.5
```

Figura 4.21: Output do script

4.1.5 Cronjobs

Na industria da tecnologia, os *cronjobs* têm uma enorme importância e apresentam uma utilização elevada, visto serem fundamentais para fazer a gestão automática de tarefas ou comandos periódicos.

Neste trabalho, a utilização de *cronjobs* tem o objetivo de, periodicamente, fazer um pedido à API do Instituto Português do Mar e da Atmosfera sobre dados referentes ao risco de incêndio, probabilidade de precipitação, temperatura mínima e máxima.

Para tal ser possível, é necessário editar o ficheiro *crontab* que está presente no diretório */etc*, ou seja, */etc/crontab*. Este ficheiro é um arquivo de configuração de todo o sistema usado pelo daemon Cron, sendo este o responsável por agendar e executar tarefas ou comandos em intervalos definidos pelo utilizador ou administrador, conhecidos como *cron*.

Na imagem seguinte é possível visualizar o ficheiro e a alteração feita ao mesmo.

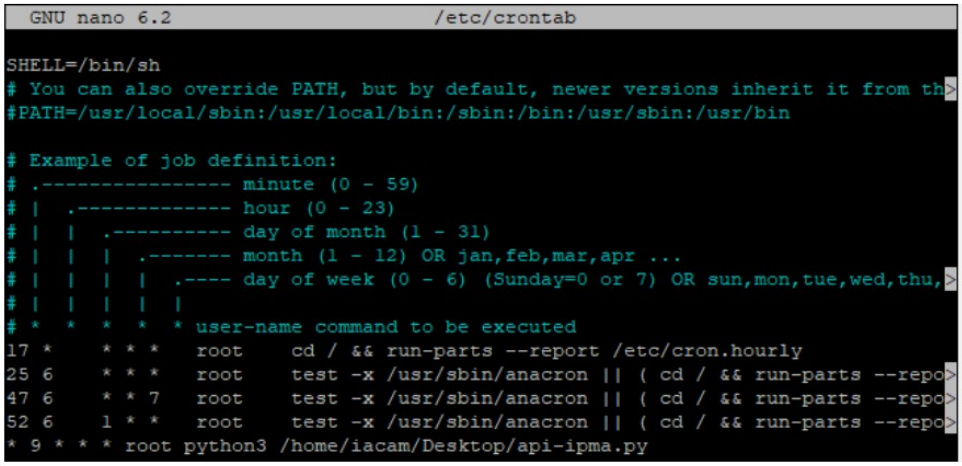


Figura 4.22: Ficheiro /etc/crontab

O comando adicionado foi "*** 9 * * * root python3 /home/iacam/Desktop/api-ipma.py**"que tem como objetivo, executar o script "*api-ipma.py* para fazer os pedidos à API do IPMA e, após isso, enviar os dados com recurso ao protocolo MQTT para o context broker Mosquitto.
O formato da definição de um "*job*"tem três campos, um deles diz respeito à configuração do tempo, o segundo campo diz respeito ao utilizador e, por último, o comando ou tarefa a ser executado. Neste caso, o script foi configurado para ser executado todos os dias às 09 horas da manhã e tem a permissão de root para executar o comando.

4.2 Dificuldades encontradas

No que diz respeito aos problemas encontrados durante todo o desenvolvimento do projeto, foram registados e anotadas as abordagens de resolução, sendo descritos nesta secção. Apesar de todos os esforços, nem todos os problemas acabaram resolvidos no decorrer do projeto.

4.2.1 Requisitos Fiware e Docker

A utilização do *Core Context Management: The NGSI-v2 Interface*, requer o cumprimento de um conjunto de pré-requisitos que inclui, no caso do Linux, um sistema operativo Ubuntu 22.04.2 LTS.

Também é necessário utilizar o software Docker que igualmente apresenta os seus pré-requisitos, tais como um kernel de 64-bit e um CPU que suporte virtualização, suporte de virtualização KVM, o QEMU deve ser a versão 5.2 ou a mais recente e a máquina, no mínimo, ter 4 GB de memória RAM.

Neste caso,o problema que ocorreu é que, inicialmente, a versão utilizada do Ubuntu era uma versão anterior à que está presente nos pré-requisitos. Também a memória alocada à máquina virtual que suporta o desenvolvimento do projeto tinha menos

que o recomendado. Por fim, o *virtual hard disk* tinha pouco espaço, apenas 20GB, sendo que por padrão, cada *container* é configurado para ter 10 GB de tamanho de disco.

Solução Este problema foi resolvido através da criação de uma máquina virtual nova com as seguintes especificações:

- Sistema operativo - Ubuntu 22.04.2 LTS,
- Capacidade do hard disk de 50 GB,
- Memória RAM 6GB e
- Quatro processadores.

Esta modificação foi suficiente para ultrapassar o problema e tornar possível a continuação do desenvolvimento do projeto. Na Tabela 4.1 são descritas as especificações anteriores à resolução do problema e depois da aplicação da solução.

Tabela 4.1: Comparação das características da máquina

Características	**Antes**	**Depois**
Sistema Operativo	Ubuntu 18.04.6 LTS	Ubuntu 22.04.2 LTS
Capacidade hard disk	20GB	50GB
Memória RAM	2GB	6GB
Processadores	1	4

4.2.2 Aquisição de equipamento

Como para a realização do projeto são necessários alguns componentes físicos, consequentemente, é necessário adquiri-los.

Neste campo, o primeiro problema foi na aquisição da Raspberry Pi que, infelizmente não foi possível, logo foi utilizado um outro componente para a substituir e ser capaz de fazer a sua função no projeto, que passava por ser o servidor principal que seria responsável pela *framework* FIWARE, broker mosquitto e a aplicação web. No que toca aos sensores e atuadores, referenciados em 4.1.1, também foi necessário adquiri-los. Como este projeto não é financiado, a aquisição destes equipamentos

foi feita individualmente, tendo demorado o seu tempo na procura de fornecedores e na entrega dos mesmos.

Solução A solução encontrada passou pela criação de uma máquina virtual que suporta todo o core do sistema desenvolvido, isto é, suportar a aplicação web, aplicação FIWARE, broker mosquitto, entre outros.

Ainda dentro deste problema, a aquisição dos sensores e atuadores foi um imprevisto que surgiu, mas teve uma solução rápida em alguns equipamentos com a exceção dos componentes relés e sensor de gás, que apresentaram um tempo de entrega fora do previsto.

4.2.3 Scripting

No que diz respeito ao *scripting*, o problema que surgiu foi que inicialmente esta componente estava a ser realizada num máquina virtual que é uma emulação de um computador físico. Sendo que cada máquina virtual atua como um ambiente independente e isolado com o seu próprio hardware virtualizado, incluindo processador, memória, armazenamento e interfaces de rede. Logo, estava a ser impossível obter os dados pretendidos, isto é, dados como temperatura dos CPU's, devido à situação de virtualização.

Solução Como, inicialmente, a parte do *scripting* estava a ser desenvolvida numa máquina virtual, a obtenção de dados dos sensores (e.g. a temperatura) eram impossíveis devido à virtualização. Então, a solução encontrada foi substituir essa máquina virtual por um servidor físico em que a obtenção dos dados já é possível.

4.2.4 Configuração do FIWARE

Foi neste tópico que o aparecimento de diversos problemas foi mais acrescido, na qual houve problemas de configurações, implementação, entre outros.
Como já foi referido anteriormente, é no ficheiro de configuração *docker-compose.yml* que estão presentes todas as configurações dos serviços utilizados, bem como a atribuição de portas, rede, imagens, dependências, ambiente, entre outros.
Para ser possível a comunicação entre os dispositivos IoT com a aplicação FIWARE, são necessários serviços como o protocolo MQTT, o agente IoT *ultra-light* e o Orion.
Um dos problemas que surgiu no desenvolver do projeto foi a correta atribuição das portas para possibilitar a comunicação, contudo o micro controlador ESP32 não obteve sucesso ao comunicar com estes serviços.
Onde surgiu outro problema foi na implementação do software Docker, que foi uma tarefa árdua e trabalhosa devido aos seus requisitos que também já foram referidos.
E, por fim, ainda dentro do mesmo software, não foi possível utilizar a versão *desktop* do Docker pois sempre que era inicializado o erro que surge é "**Docker Engine stopped**".

Solução A questão da implementação do Docker, apesar de ter sido um processo árduo, foi superado com sucesso. Relativamente ao Docker Desktop, como este não estava operacional, a solução foi utilizar o Docker CLI que é menos amigável e prático, mas que cumpre a sua função e foi a solução encontrada para tornar o desenvolvimento do projeto possível.

Por resolver No entanto, a parte da configuração do Context Broker, apesar de se verificar todas as configurações, não foi possível obter uma comunicação positiva entre o ESP32 com o Broker MQTT. Uma possível causa, é a falta de permissões do lado do servidor em aceitar certos endereços e dispositivos, rejeitando assim os mesmos, uma vez que os erros que obtidos foram "**rc = -2**"que indica uma falha de conexão (*MQTT_CONNECT_FAILED*) e "**rc = -4**"que informa que o tempo limite de conexão foi atingido (*MQTT_CONNECTION_TIMEOUT*).

4.2.5 Aplicação web

Inicialmente, é necessário fazer uma breve introdução do que é o *Cross-origin Resource Sharing* (CORS). O CORS é um mecanismo utilizado pelos diversos *browsers* com o intuito de compartilhar recursos entre diferentes origens, sendo o mesmo uma especificação do World Wide Web Consortium (W3C) que faz uso de *headers* do protocolo HTTP para informar os *browsers* se determinado recurso pode ou não ser acedido.

O processo de CORS passa por algumas fases que estão presentes no diagrama apresentado seguidamente.

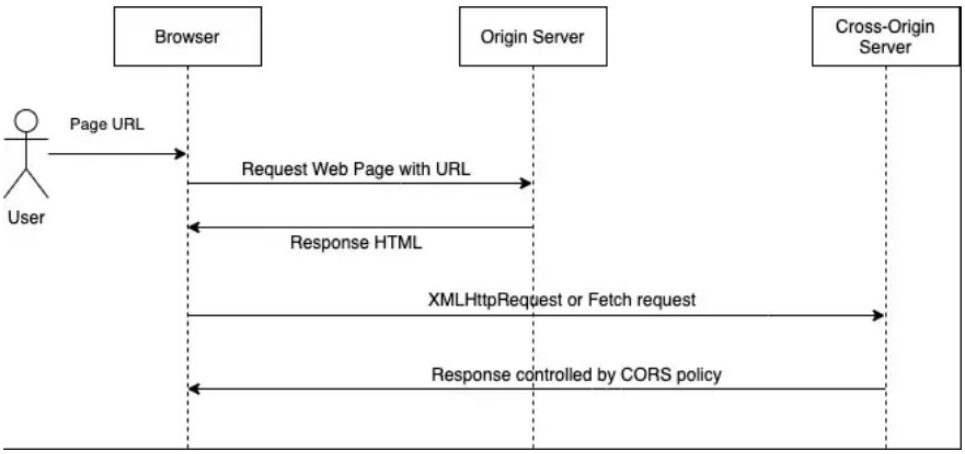


Figura 4.23: CORS

Contudo, podem surgir erros associados, ou seja, erros de CORS. Sendo esses erros como:

- XMLHttpRequest ou Fetch API do Javascript;
- Web fonts;
- WebGL textures e

- CSS Shapes.

No caso do projeto, o erro de *CORS* que surgiu foi dentro do tópico de XMLHttpRequest, ou seja, neste caso é sobre a falta de *headers* por parte do servidor.
A aplicação web pretende acessar a informação que se encontra na plataforma FIWARE, para preencher campos com essa mesma informação. Contudo, para isso ser possível, é necessário fazer um pedido *GET*. Sendo que, do lado do servidor, é necessário configuração para aceitar esses pedidos, isto é, é necessário adicionar *headers* para aceitar.

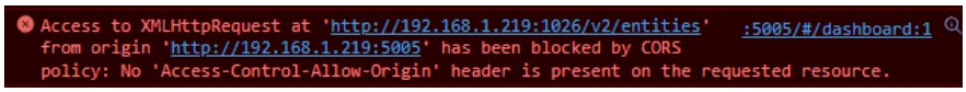


Figura 4.24: CORS

Por resolver Uma possível solução passaria por tentar implementar os *headers* do lado do servidor, algo que foi testado contudo sem sucesso.

4.2.6 Sensores e Atuadores

Após a aquisição de alguns componentes, sendo eles sensores e atuadores, foi percetível que alguns equipamentos vieram danificados ou avariados. Pois, apesar de todos os testes possíveis, a nível de *firmware*, que são detalhados seguidamente os componentes continuaram não operacionais.
Fazem parte desse problema o detetor de chama e fonte de calor e um dos canais do relé.

Por resolver Como se trata de um problema no próprio sensor/atuador, a única solução seria proceder à sua substituição por um equipamento novo e que estivesse operacional. Esta solução não foi possível em tempo útil.

4.3 Desvios em relação ao plano inicial

Relativamente ao plano inicial, houve dois desvios com alguma relevância, mas que não prejudicaram ou influenciaram o desenvolvimento do projeto.

O primeiro desvio refere-se à indisponibilidade no mercado para aquisição de um micro-computador Raspberry Pi, que acabou sendo substituído por uma máquina virtual. Porém, o desvio resultante acabou por não alterar significativamente o desenvolvimento do projeto.

O segundo desvio decorreu do plano inicial de prototipar dois dispositivos IoT. Infelizmente, não foi possível adquirir sensores, atuadores e outros componentes para dois dispositivos, resultando no desenvolvimento um protótipo com apenas um dispositivo IoT.

Capítulo 5

Avaliação crítica e trabalho futuro

Neste capítulo são identificados alguns aspetos que poderão conduzir a trabalho futuro, quer pelo aperfeiçoamento de componentes do trabalho desenvolvido, quer pelo desenvolvimento de novos componentes que adicionem valor ao sistema.

5.1 Aplicação Web

Relativamente à aplicação web, este é um campo em que existem alguns aspetos que poderiam ser melhorados e, outros, implementados.

5.1.1 Melhorias

A aplicação web foi resultado de uma implementação em simultâneo com o estudo da *framework* de desenvolvimento Angular. Desta forma, pode não ter sido desenvolvida com práticas que permitam otimizações, com impacto no desempenho da aplicação. Desta forma, uma revisão à aplicação web poderá melhorar o desempenho em algumas das suas funcionalidades.

A interface de utilizador também poderá ser sujeita a melhorias, de forma a torná-la mais amigável e intuitiva para o utilizador final.

Por fim, seria necessário resolver o problema dos CORS (mencionado no Capítulo 4, o mecanismo de *browser* que permite acesso controlado a recursos localizados fora de um determinado domínio.

5.1.2 Novas funcionalidades

No que toca à adição de novas funcionalidades, num primeiro ponto de vista, seria interessante idealizar a implementação de uma plataforma multi-dispositivo, ou seja, a plataforma ser capaz de monitorizar diversos dispositivos simultaneamente. Tornando possível, a visualização dos estados dos atuadores de dispositivos diferentes e manipular o seu estado se o administrador ou consumidor final assim o pretender.

Outra funcionalidade que pode enriquecer este projeto, passa uma abordagem multi-utilizador também com o intuito de melhorar a segurança da plataforma e permitir a diferentes utilizadores usufruir da plataforma, não esquecendo fazer a sua distinção com privilégios diferentes.

Por fim, uma outra componente que encaixa no trabalho futuro são os alertas na aplicação web. Os mesmos são utilizados para alertar os administradores da plataforma de possíveis erros ou falhas nos sensores e na criticidade do ambiente interior nos bastidores. Para isso, poderia optar pela utilização de *WebSockets*, que é uma tecnologia capaz de criar uma ligação entre o cliente e o servidor e permitir a comunicação entre eles em tempo real. Já o HTTP permite receber dados sem ter que enviar um pedido separado e uma vez estabelecida a ligação, os dados fluem por si mesmos sem necessidade de enviar o pedido. (Taylor 2022)

5.2 Firmware

Entrando agora na secção do *firmware*, o mesmo também necessita de melhorias em diferentes campos. Adicionalmente, também poderiam ser implementadas algumas novas funcionalidades.

5.2.1 Melhorias

Sobre as melhorias que o *firmware* pode apresentar, uma relevante passa pelo algoritmo de tomada de decisão, dado que, neste momento, o mesmo tem as probabilidades fixas. Logo, caso um ou mais sensores apresentem alguma falha ou avaria, o mesmo não se consegue adaptar a essa realidade e apenas é capaz de avisar os administradores que existe um erro no sensor em questão e o tipo de erro (má leitura ou de falha na deteção do dispositivo).

Outro tópico é a comunicação com o *context broker* MQTT presente na aplicação FIWARE. Apesar do *container* estar em execução, não foi possível estabelecer a comunicação entre o mesmo e o dispositivo IoT. O problema poderá ser ao nível das permissões do lado da aplicação FIWARE em que esta não permite ao dispositivo se conectar ao *context broker* ou também pode ser um erro de versões, isto é, a versão que está a ser executada na aplicação FIWARE ser uma versão que não seja compatível com a que está implementada no dispositivo IoT.

5.2.2 Novas funcionalidades

No que diz respeito a novas funcionalidades, seria relevante pormenorizar a componente de erro. Caso um sensor esteja comprometido na sua função, a consequente tomada de decisão deverá ter esse o tipo de erro reportado em conta na atualização do grau de confiança desse sensor.

Um segundo tópico seria relacionado com a melhoria do *desempenho*, através de uma análise do código verificar se seria possível otimizar o *firmware*.

Capítulo 6

Conclusão

O conceito de controlo e monitorização do interior de uma sala de servidores reveste-se de uma importância elevada pois, caso o ambiente derive para uma situação não-nominal, aumenta a probabilidade de um incidente com impacto na capacidade operacional da organização.

Este projeto teve como objetivo avaliar a possibilidade de construir um sistema de monitorização, controlo e proteção de uma sala de servidores, recorrendo a tecnologias não-proprietárias. A utilização da plataforma FIWARE, foi fundamental para a realização do projeto, servindo como middleware entre a aplicação web (consumidora dos dados amostrados pelos sensores) e os dispositivos IoT que recolhem esses dados ou que atuam no ambiente da sala dos servidores. Esta *framework* permite integrar, com razoável facilidade, tecnologias com características heterogéneas, sobretudo no lado ciberfísico onde se encontram os elementos ligados ao ambiente físico (nomeadamente sensores e atuadores), enquanto fornece simultaneamente uma plataforma para construção de aplicações que tanto consomem o contexto do ambiente monitorado, e/ou que decidem sobre as medidas a tomar para influenciar esse mesmo ambiente.

Contudo, a integração do *broker* MQTT e do respetivo *FIWARE context broker* revelou uma série de contratempos, que nem todos conseguiram ser resolvidos. Esta foi uma das limitações mais sérias do trabalho realizado.

No desenvolvimento do projeto IACAM, foi utilizada a *framework* Angular que se revelou uma tecnologia de desenvolvimento positiva. Este *framework* é bastante intuitiva e de fácil aprendizagem; adicionalmente, dispõe de uma comunidade ativa e extensa documentação.

Este projeto apresenta várias limitações, sendo que os resultados não são totalmente satisfatórios. Mais trabalho seria necessário para ultrapassar as todas dificuldades. A impossibilidade de adquirir todos os componentes desejáveis condicionou o trabalho, apesar dos esforços de mitigação do impacto no desenvolvimento do trabalho.

Também teria sido desejável construir um protótipo mais próximo de um caso prático real, que incluísse uma sala com um bastidor com equipamentos informáticos e de rede, e atuadores reais, como ventiladores que efetivamente modificassem as características do ambiente quando solicitados pelo sistema. A implantação do protótipo num ambiente mais próximo da realidade permitiria adquirir dados reais e realizar

uma análise para estabelecer uma linha de base. Esta análise seria fundamental para o ajuste dos parâmetros de controlo para se obter uma solução à medida de cada instalação; previsivelmente, poderiam ser treinados modelos para a utilização de inteligência artificial no controlo de instalações.

Apesar das dificuldades em integrar as diversas tecnologias de forma a responder ao problema inicialmente proposto, que resultaram em algumas limitações já descritas, é possível antever que é possível construir sistemas de monitorização, controlo e proteção das instalações de TI recorrendo a tecnologias não-proprietárias.

Bibliografia

3M (2023). *Extinção de incêndios Novec 1230*. url: https://www.3m.com.pt/3M/pt_PT/novec-pt/como-utilizar/extinguir-incendios/.

AKCP (2021). *Server Rack Monitoring System - Data Center Remote Monitoring*. Accessed: 2023-07-21. url: https://www.akcp.com/articles/server-rack-monitoring-system-what-it-is-and-why-its-important/.

– (2023). *AKCP Remote Sensor Monitoring | Data Center Monitoring*$_2$023. Accessed: 2023-07-21. url: https://www.akcp.com/akcp-products/akcpro-server/.

Arduino (2023). *Software*. url: https://www.arduino.cc/en/software.

Critérios de Design Ambiental (mar. de 2021). url: https://www.ibm.com/docs/pt-br/power7?topic=POWER7%2Fp7ebe%2Fp7ebetempandhumiditydesign.html.

Docker (2003). *What is Docker?* Accessed: 2023-09-13. url: https://aws.amazon.com/pt/docker/.

embedded, Explore (2023). *Overview of ESP32 features*. url: https://www.exploreembedded.com/wiki/Overview_of_ESP32_features._What_do_they_practically_mean%3F.

English, Cambridge International Dictionary of (2023). *"IT Definition" Cambridge International Dictionary of English*. url: https://dictionary.cambridge.org/dictionary/english/information-technology.

ESPRESSIF (2023). *Development Boards | Espressif Systems*. url: https://www.espressif.com/en/products/devkits.

FIWARE (set. de 2023). *FIWARE*. url: https://www.fiware.org/catalogue/.

IBM (2023a). *IBM Cloud Docs*. Accessed: 2023-07-23. url: https://cloud.ibm.com/docs/overview?topic=overview-understanding-dr.

– (2023b). *IBM Sterling B2B Integration SaaS*. Accessed: 2023-07-23. url: https://www.ibm.com/docs/en/b2bis?topic=overview-disaster-recovery.

– (2023c). *What is blockchain for business?* url: https://www.ibm.com/topics/blockchain-for-business.

Information Technology (jun. de 2018). url: https://www.encyclopedia.com/science-and-technology/computers-and-electrical-engineering/computers-and-computing/information-technology.

Jones, Penny (mar. de 2013). *Overheating brings down Microsoft Data Center*. url: https://www.datacenterdynamics.com/en/news/overheating-brings-down-microsoft-data-center/.

Jstrom99 (out. de 2019). *File:RaspberryPi 4 Model B.svg*. url: https://commons.wikimedia.org/wiki/File:RaspberryPi_4_Model_B.svg.

LastMinuteEngineers (jul. de 2022). *Configuring and Handling ESP32 GPIO Interrupts In Arduino IDE*. Accessed: 2023-08-07. url: https://lastminuteengineers.com/handling-esp32-gpio-interrupts-tutorial/.

McFarlane, Robert (mai. de 2022). *Uninterruptible Power Supply - definition from techtarget.com*. url: https://www.techtarget.com/searchdatacenter/definition/uninterruptible-power-supply.

Meier, Stephanie (jul. de 2022). «How does technology drive business transformation?» Em: *IBM Business Operations Blog*. url: https://www.ibm.com/blogs/internet-of-things/how-does-technology-drive-business-transformation/.

Microsoft (nov. de 2019). *Microsoft Hybrid Cloud for Enterprise Architects*. url: https://learn.microsoft.com/en-us/office365/enterprise/media/Hybrid-Poster/MSFT_cloud_architecture_hybrid.pdf.

– (nov. de 2021). *Visual studio code frequently asked questions*. url: https://code.visualstudio.com/docs/supporting/FAQ.

Neenan, Sarah e Stephen J. Bigelow (jul. de 2023). *What is hybrid cloud? the ultimate guide: TechTarget*. url: https://www.techtarget.com/searchcloudcomputing/definition/hybrid-cloud.

P2i (2019). *FM-200 (HFC-227ea)*. url: https://www.p2i.pt/fm-200/.

Pi, Raspberry (2023). *Raspberry Pi 4 Model B*. url: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/.

Sverdlik, Yevgeniy (abr. de 2015). *Ashrae's pursuit of lower humidity in data centers*. url: https://www.datacenterknowledge.com/archives/2015/04/20/data-center-cooling-ashrae-to-widen-humidity-envelope.

Taylor, Amalia (jun. de 2022). *O que são WebSockets e Como Criá-los?* url: https://appmaster.io/pt/blog/o-que-sao-websockets-e-como-cria-los.

ThingSpeak (2023). *Learn More - ThingSpeak IoT*. url: https://thingspeak.com/pages/learn_more.

Vertiv (2023). *What's Your Maximum Server Room Temperature?* url: https://www.vertiv.com/pt-latam/about/news-and-insights/articles/educational-articles/whats-the-maximum-temperature-of-your-server-room/.

Wallarm (mar. de 2023). *CoAP Protocol: What is Meaning, Architecture and Function?* url: https://www.wallarm.com/what/coap-protocol-definition.

Wang, Zeyu et al. (2022). «Business Innovation based on artificial intelligence and Blockchain technology». Em: *Information Processing Management* 59.1, p. 102759. issn: 0306-4573. doi: https://doi.org/10.1016/j.ipm.2021.102759. url: https://www.sciencedirect.com/science/article/pii/S0306457321002405.

Wickel, Alexander (2008). «Fire extinguisher systems in the data centre». Em: pp. 1–21. url: http://www.rittal.de/downloads/rimatrix5/cooling/WP_Fire_extinguish_DC_en.pdf.