André Branco Rosado

Licenciado em Ciências da
Engenharia Eletrotécnica e de Computadores

Impacto energético de um incêndio numa rede de sensores IoT

Dissertação para obtenção do Grau de Mestre em

Engenharia Eletrotécnica e de Computadores

Orientadora: Doutora Anikó Katalin Horváth da Costa,
Professora Associada com Agregação,
Universidade Nova de Lisboa - Faculdade de Ciências e Tecnologia

Júri

Presidente: Doutor Rui Miguel Henriques Dias Morgado Dinis, FCT/UNL
Arguente: Doutora Ana Inês da Silva Oliveira, FCT/UNL

NOVA SCHOOL OF SCIENCE & TECHNOLOGY

Setembro, 2021

Impacto de um incêndio numa rede de sensores IoT

1. *A robot may not injure a human being or, through inaction, allow a human being to come to harm*

2. *A robot must obey the orders given it by human beings except where such orders would conflict with the First Law*

3. *A robot must protect its own existence as long as such protection does not conflict with the First or Second Law*

- Isaac Asimov, 1942

Resumo

Na atualidade, e cada vez mais no futuro, a humanidade confronta-se com desastres. O impacto na sociedade origina, para além da eventual perda de vidas humanas, prejuízos incalculáveis. No entanto, com a imparável marcha da evolução tecnológica, é possível conceber os sistemas ciber-físicos cada vez mais capazes de responder a qualquer tipo de desastre no sentido de previsão, preparação, e resposta durante o evento, tornando assim um determinado edifício ou zona mais segura. Para tal é essencial existir uma estrutura de Gestão de Risco de Desastre bem concebida.

O foco do trabalho consiste na simulação de dois cenários distintos num edifício: um estado normal e um cenário de incêndio. Em ambos, os sensores são iguais e os cenários partilham a mesma topologia, sendo, no entanto, alterados o valor da temperatura e a sua frequência de leitura. Esta dissertação contribui para a comunidade de investigação com uma pesquisa profunda em torno do impacto de um desastre e os planos de evacuação, particularmente para a iniciativa *safe escape* através de uma estrutura proposta, assim como a introdução ao simulador *cooja*, onde é realizada a análise de uma rede de sensores IoT e simulados os cenários. Inicialmente constatou-se que o protocolo indicado é o RPL, entre outras características base da rede de sensores, como distribuição dos sensores pelo edifício de um modo estratégico. Após a comparação dos resultados de simulação, conclui-se o forte impacto energético que um incêndio pode ter numa rede de sensores para efeitos de comunicação.

A par de toda a comunidade de investigação, o objetivo comum passa por tornar os edifícios locais mais seguros para as pessoas através da melhoria dos sistemas ciber-físicos. Assim, a grande demanda é sem dúvida, conceber melhores sistemas ciber-físicos, para que estes nos ofereçam segurança.

Palavras-chave: Sistemas ciber-físicos, Segurança, Simulação, IoT

Abstract

Nowadays, and increasingly in the future, humanity is facing disasters and calamities with severe impacts on society that cause in addition to the possible loss of human lives, incalculable damage. However, with the unstoppable advances of technological evolution, it is possible to conceive cyber-physical systems which are increasingly capable of responding to any type of disaster, and with an ability to respond with greater efficiency, in the direction of forecasting, preparation, and response during the event, thus making any building or zone safer. For this it is essential to have a well-structured disaster framework.

The focus of the work consists in simulating two different scenarios in a building: a normal state and a fire scenario. In both, the sensors are the same and the scenarios share the same topology, but the temperature value and its reading frequency are changed. This dissertation contributes to the research community with in-depth research around the impact of a disaster and evacuation plans, particularly for the *safe escape* initiative through a proposed framework, as well as the introduction to the *cooja* simulator, where the analysis of a IoT sensor network is performed and the scenarios are simulated. Initially it was found that the protocol indicated is RPL, among other base characteristics of the sensor network, such as distribution of the sensors throughout the building in a strategic way. After comparing the simulation results, it is concluded the strong energy impact that a fire can have on a sensor network for communication purposes. Along with the entire research community, the common goal is to make local buildings safer for people by improving cyber-physical systems.

Therefore, the great demand is to design better cyber-physical systems to guarantee our safety.

Keywords: Cyber-Physical Systems, Safety, Security, Simulation, IoT

Índice

Lista de Figuras

Lista de Tabelas

Acrónimos

2G Segunda Geração.

3G Terceira Geração.

5G Quinta Geração.

6LoWPAN *IPv6 over Low power Wireless Personal Area Networks*.

AODV *Ad-hoc On Demand Distance Vector*.

AVAC Aquecimento, Ventilação e Ar Condicionado.

BAS *Building Automation System*.

CCRP *Capacity Constrained Route Planner*.

CPS *Cyber-Physical Systems*.

CPU *Central Process Unit*.

DAG *Directed Acyclic Graph*.

DODAG *Destination Oriented DAG*.

EA *Earliest Arrival time*.

ETX *Expected Transmission Count*.

GRD Gestão de Risco de Desastre.

HMI *Human-Machine Interface*.

ICMP *Internet Control Message Protocol*.

IEEE *Institute of Electrical and Electronics Engineers*.

IoE *Internet of Everything*.

IoT *Internet of Things*.

IP *Internet Protocol*.

IPv6 *Sexta versão de Internet Protocol*.

LOADng *Lighweight On-demand Ad hoc Distance-vector Routing Protocol – Next Generation*.

LPM *Low Power Mode*.

LTE *Long Term Evolution*.

OMM Organização Meteorológica Mundial.

PIB Produto Interno Bruto.

RAM *Random Acces Memory*.

RERR *Route Error*.

RFID *Radio Frequency Identification*.

RGPD Regulamento Geral de Proteção de Dados.

ROM *Read Only Memory*.

RPL *Routing Protocol for Low Power and Lossy Networks*.

RREP *Route Reply*.

RREQ *Route Request*.

SCADA *Supervisory Control And Data Acquisition*.

sht *Sensor Humidity Temperature*.

SMA *Simple Moving Average*.

TCP *Transmission Control Protocol*.

TTL *Time To Live*.

UDP *User Datagram Protocol*.

UNESCO *United Nations Educational, Scientific and Cultural Organization*.

USB *Universal Serial Bus*.

Wi-Fi *Wireless Fidelity*.

Capítulo 1

Introdução

1.1 Motivação

Todos os anos os desastres naturais, bem como os causados pelo ser humano, têm um forte impacto na humanidade, sendo os mais comuns terramotos, cheias, incêndios e ciclones. Além do seu alto nível de devastação e prejuízo, têm vindo a ser cada vez mais frequentes. Isso deve-se, em parte, ao rápido desenvolvimento económico e consequente crescimento urbano, o que por sua vez expõe qualquer cidadão a estes eventos. Por outro lado, crê-se que as alterações climáticas também sejam responsáveis por uma maior frequência e intensidade destes eventos devastadores.

Uma catástrofe pode afetar diretamente pessoas, edifícios ou até mesmo património histórico. Para o ser humano existem situações mais irreversíveis. Além de perder a casa ou bens, mais concretamente perder algum familiar, pode-se ainda enfrentar o perigo de morte ou danos físicos.

Tem um maior impacto nos chamados países em desenvolvimento, pois estes tem menos meios de resposta, e a sua população não está tão bem preparada. Pretende-se assim reduzir significativamente o número de mortes e de pessoas afetadas, e reduzir também os prejuízos causados a empresas ou países. É também importante tentar conservar o património histórico, de modo a não afectar a cultura. Visto este, ser quase sempre posto em segundo plano, pois o primeiro instinto é salvar vidas e propriedades.

Desastres, geralmente, são o resultado da combinação de vulnerabilidades com a imprevisibilidade da mãe natureza. Através de várias soluções e inovações, pretende-se otimizar os sistemas ciber-físicos, de modo a preparar melhor os edifícios e a população para estes eventos, esta tese aborda essencialmente o estudo de uma rede de sensores assim como o impacto de um incêndio na mesma.

1.2 Enquadramento

Os sistemas ciber-físicos já fazem parte do nosso dia a dia, integrando diferentes tipos de estrutura ou processo, com o mais variado tipo de aplicações. Poder aliar qualquer processo fisíco, à capacidade computacional, trouxe sem dúvida uma série de vantagens e um leque de novas possibilidades.

É interessante, além de todas as vantagens existentes, poder preparar estes sistemas para qualquer tipo de desastre. Pretende-se então, combinar os planos tradicionais de evacuação já adotados, com algoritmos que se adaptem a diferentes cenários. Ou seja, eventualmente os sensores integrantes destes sistemas, presentes na maior parte dos edifícios com os padrões básicos de segurança, sincronizados com um algoritmo, que tem a capacidade de calcular, em tempo real, o melhor caminho possível para a saída, ou para algum local mais seguro do edifício. Para tal, o trabalho desenvolvido nesta dissertação aborda as redes sensores, desde a criação da sua topologia, e terminando no impacto energético de um incêndio.

É de notar, que nem sempre o caminho mais rápido é o mais seguro.

1.3 Objetivos

Com esta tese, pretende-se numa primeira fase estudar a análise de risco de qualquer edifício ou espaço, ou seja, identificar as vulnerabilidades que pudessem originar num desastre, e os recursos ao dispor para combater esse tipo de eventos. Para isso, há que ter em conta toda infraestrutura do edifício.

Posteriormente, será analisado em detalhe um caso prático, este caso prático consiste num edifício e na disposição de uma rede de sensores. Esta tese, pretende demonstrar que é possível elevar os sistemas ciber-físicos para outro nível, dando um forte contributo na gestão de desastres. No futuro, será possível analisar diferentes cenários, de um modo autónomo, e escolher os caminhos mais seguros para a saída de um edifício. Existirão sempre desastres, no entanto, o principal objetivo é reduzir o seu impacto.

Numa fase final, é preciso garantir a segurança destes sistemas, assim como a privacidade de quem o rodeia. Pois, ataques cibernéticos têm vindo a crescer exponencialmente, e além de comprometer os sistemas ciber-físicos, podem também pôr em causa a integridade física de seres humanos, além disso. É importante não esquecer a questão da privacidade, pois após alguns eventos que comprometeram a vida particular de cidadãos, surgiu o Regulamento Geral de Proteção de Dados (RGPD).

Estes são os principais desafios a que esta tese se propõe, permitindo assim que os sistemas ciber-físicos, para além de facilitarem a nossa vida, nos ofereçam mais segurança e em que podemos confiar.

1.4 Contributo

Esta tese tem origem no projeto *safe escape*, por sugestão da Professora Orientadora. Aqui são levantadas as principais questões, e definidos os objetivos.

- Como tornar os sistemas ciber-físicos mais seguros ?
  - Reduzir o número de mortes e pessoas afetadas em desastres
  - Aumentar a resiliência destes sistemas face a situações catastróficas

Pretende-se assim através da análise de redes sem fios e testes extensos otimizar os planos de evacuação, dando um forte contributo para este programa.

1.5 Metodologia

Em 1.1 é representada a metodologia utilizada, de modo a esquematizar todo o processo de pesquisa, de um modo sucinto, proporcionando uma maior fiabilidade dos conteúdos apresentados.

Figura 1.1: Metodologia adotada

A pesquisa intensiva é feita essencialmente pela procura de conteúdos relacionados principalmente com segurança (desastres, planos de evacuação) e sistemas ciber-físicos, presente em repositórios online (livros, *papers* científicos, artigos, entre outros). No entanto, foi feito um questionário essencial para a validação dos dados mais adiante. Este, permite uma recolha de dados importante, em que hipoteticamente coloca uma pessoa numa situação de desastre, permitindo assim uma análise ao seu comportamento.

- **Pesquisa**: Dados secundários, pesquisa qualitativa
- **Questionário**: Dados primários, pesquisa qualitativa e quantitativa

Posteriormente é feita uma análise cuidada destes dados, e juntamente com o estudo de possíveis algoritmos a implementar, é possível chegar ao veredicto do melhor modelo a adotar.

1.6 Organização

No semestre de preparação da dissertação foi elaborado um relatório, que não é mais do que um esboço da versão final desta tese de mestrado, permitiu na altura ter uma visão de todo o trabalho realizado até à data, assim como um maior enquadramento do tema. Foi feita uma apresentação via *Zoom* onde foi apontado o trabalho futuro e recolhido algum *feedback* dos professores.
A dissertação está dividida em 7 capítulos.
O primeiro capítulo introduz e enquadra o problema abordado na dissertação. O segundo capítulo descreve o estado da arte sobre o problema abordado com uma descrição de abordagens e de trabalhos de outros investigadores que lidaram com desafios semelhantes. O terceiro capítulo aborda o tema de um modo mais detalhado, é identificado o ambiente do tema assim como o seu meio de comunicação, são referidas ainda as ferramentas utilizadas ao longo desta jornada. Ainda no fim do terceiro capítulo é introduzida uma possível estrutura de gestão de risco de desastre.
No quarto capítulo, através de um caso prático, é implementada uma rede de sensores *Internet of Things* num edifício histórico tornando este num edifício inteligente, posto isto, são analisados dois cenários e verificado o impacto de um incêndio num edifício desta categoria. Para finalizar, o quinto capítulo resume as conclusões e aborda as perspectivas de futuro.

Capítulo 2

Enquadramento Tecnológico

2.1 Sensores e sinais

A detecção de sinais é uma técnica utilizada para recolher informações sobre um determinado objecto ou processo físico, onde tem como principal objetivo o registo da ocorrência de eventos (ou seja, mudanças de estado, tais como uma queda de temperatura ou pressão). Um dispositivo que tenha capacidade para tal é chamado de sensor. Por exemplo, o corpo humano está equipado com sensores capazes de capturar informações visuais do ambiente que o rodeia (olhos), informações acústicas como sons (ouvidos) e odores (nariz) [1]. De uma perspectiva técnica, um sensor é um dispositivo que traduz parâmetros ou eventos do mundo físico em sinais que podem ser medidos e analisados.

2.1.1 Classificação e especificações

A classificação de um sensor varia de muito simples até ao complexo. Dependendo da sua aplicação, além de que podem ser adotadas diferentes abordagens. São referidos vários tipos de sensores em [2].

- Existem dois tipos de sensores: passivos e ativos. Um sensor passivo não necessita de nenhum tipo de alimentação, enquanto o sensor ativo é dependente de um circuito que emite um feixe de luz e outro que a detecta. Os sensores ativos requerem energia para o seu funcionamento, essa energia é denominada de sinal de excitação, que por sua vez é modulado pelo próprio sensor de modo a produzir o sinal de saída. Devido ao facto dos sensores passivos serem mais sensíveis, a sua instalação é geralmente feita em ambientes fechados. Já os ativos são mais indicados para o uso em ambientes externos.

- Os sensores podem ser classificados como absolutos e relativos. Um sensor absoluto detecta um estímulo com exatidão, através de uma escala física absoluta que é independente das condições de medição, enquanto um sensor relativo produz um sinal consoante a diferença de valores medidos, como é o caso do termopar, este sensor produz uma voltagem elétrica que é função da diferença de temperatura, através de dois fios. Um exemplo de um sensor absoluto é um termístor uma resistência sensível à temperatura. A sua resistência eléctrica está directamente relacionada com a escala de temperatura absoluta, em graus Kelvin.

Independentemente do tipo de sensores, existe uma série de especificações determinantes na escolha do sensor consoante a sua aplicação.

Tabela 2.1: Especificações de sensores [1]

Sensibilidade	Intervalo de estímulos
Estabilidade	Resolução
Precisão	Seletividade
Velocidade de resposta	Condições Ambientais
Características de sobrecarga	Linearidade
Histerese	*Dead band*
Tempo de vida útil	Formato de saída
Custo, tamanho e peso	Outro

Os sensores escolhidos para uma determinada aplicação, devem ter em conta as suas propriedades físicas, por exemplo, temperatura, pressão, luz ou humidade.

A Tabela 2.2, retirada de [1], resume alguns tipos de sensores e as respetivas aplicações.

Tabela 2.2: Classificação e exemplos de sensores

Tipo	**Exemplo**
Temperatura	Termístor, termopar
Pressão	Manómetros, barómetros
Ótico	Fotodíodos, fototransístores, infravermelhos
Acústico	Ressonador piezoelétrico, microfones
Mecânico	Medidores de tensão, sensores táteis
Movimento	Acelerómetros, giroscópios
Fluxo	Anemómetros, sensores de fluxo de massa de ar
Posição	GPS, sensores ultra-som, inclinómetro
Electromagnético	Sensor de efeito *Hall*
Químico	Medidor Ph, sensor electroquímico
Humidade	Sensor capacitivo e resistivo
Radiação	Detetor de ionização

2.1.2 Rede de Sensores sem fios

As redes de sensores sem fios são consideradas uma das tecnologias mais importantes do século XXI [3]. Graças aos recentes avanços dos sistemas microeletrónicos e das tecnologias de comunicação sem fios, os sensores de tamanho reduzido, baratos e inteligentes deram origem à possível implementação de uma rede de sensores que comunicam entre si através de ligações sem fios e da Internet, abrindo a porta para inúmeras aplicações.

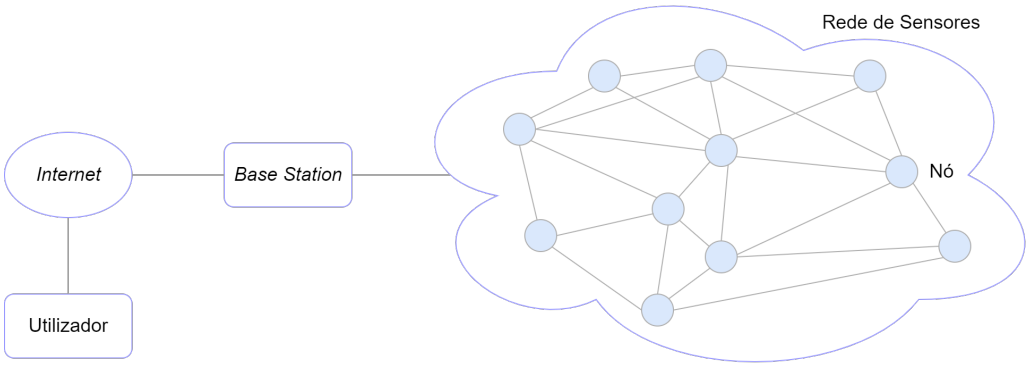

Figura 2.1: Topologia de rede de sensores

Uma rede de sensores sem fios, normalmente, consiste num elevado número de nós de baixo custo, baixa potência e multi-funcionais, que são implantados numa determinada zona ou região de interesse. Estes nós são de dimensão reduzida e, para além de sensores, são também equipados com microprocessadores embutidos e emissor-recetor de rádio. Ou seja, não só têm capacidade de detecção, assim como de processamento de dados e comunicação. De um modo mais detalhado, cada nó sensor pode ser composto por seis partes:

1. **Microprocessador:** Baixa potência e processamento de pequenas tarefas;
2. **Sensor:** Para detecção de sinais;
3. **Memória:** Para guardar programas e dados;
4. ***Transceiver*:** Emite e recebe dados sem fios;
5. **Energia:** Geralmente alimentados por pilhas ou bateriais recarregáveis;
6. **Sistema Operativo:** Existem sistemas operativos indicados para redes de sensores sem fios como por exemplo Contiki OS, Tiny OS, FreeRTOS.

Este tipo de redes comunicam a curta distância através de tecnologias de comunicação sem fios e atua como uma rede colaborativa para realizar uma tarefa comum, como por exemplo, a monitorização de ambiente, podendo alertar para alguma anomalia [4].

Em comparação com as tradicionais redes de comunicação sem fio, como as redes celulares, as redes de sensores têm particularidades únicas, nomeadamente:

- O número de nós numa rede de sensores pode ser muito maior em relação ao número de nós numa rede celular;
- Os nós são densamente instalados;
- A topologia de uma rede de sensores muda muito frequentemente;
- Geralmente, numa rede de sensores, os nós usam comunicação *broadcast*, enquanto a maioria das redes celulares são baseadas em comunicações ponto-a-ponto;
- Os nós são limitados em potência, capacidade computacional e memória, e além disso, propícios a falhas;
- Os nós podem ser identificados através de um *ID*, devido ao grande número de sensores e à sobrecarga associada.

2.1.3 Sumário

Os sensores sem fios têm vantagens significativas em relação aos sensores convencionais. Possuem a vantagem de poder ser instalados em qualquer tipo de ambiente, especialmente em casos onde seria impossível montar a mesma rede, mas com sensores convencionais, por exemplo, em terrenos inóspitos, campos de batalha, espaços exteriores ou oceanos profundos. As redes de sensores sem fios surgiram originalmente através de aplicações militares, nomeadamente vários tipos de sistemas de vigilância. No entanto, o acesso aos sensores sem fios tem sido facilitado devido ao seu baixo custo de produção, o que resultou numa maior adoção de redes de comunicação sem fios, ao invés das redes de comunicação de sensores convencionais. Como método de recolha de informação, as redes de sensores sem fios permitem ter sistemas de informação e comunicação que visam melhorar significativamente a fiabilidade e eficiência do IoT. Comparativamente com as soluções com fios, os dispositivos das redes de sensores sem fios são mais flexíveis e programáveis. Com o rápido desenvolvimento da tecnologia de sensores, as redes de sensores sem fios têm sido a tecnologia chave do IoT.

2.2 IoT

O conceito da Internet das Coisas (IoT) tem ganho cada vez mais relevância. Estima-se que entre os anos de 2018 e 2019, segundo o site *statista* [5], o número de dispositivos ativos conectados à Internet tenha ultrapassado a população mundial em número.

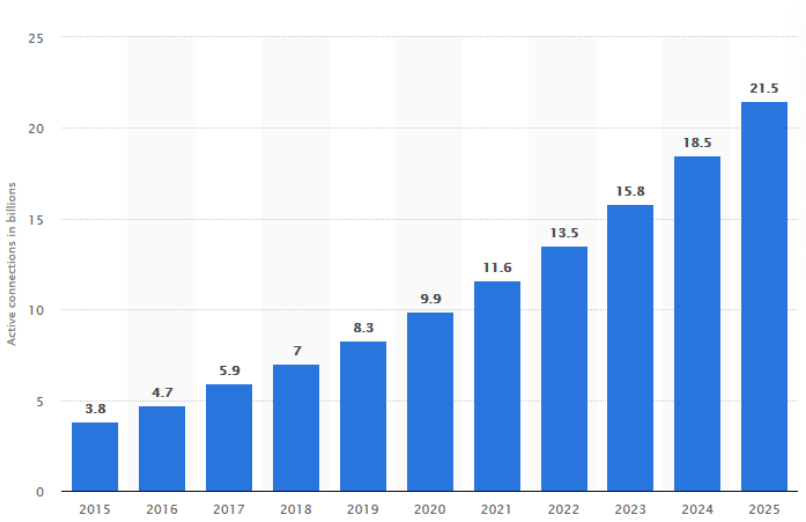

Figura 2.2: Estimativa de dispositivos IoT ativos entre 2015 e 2025

Estes números astronómicos devem-se ao enorme potencial da IoT, pois este projeta a sociedade para um futuro onde tudo se conecta à internet e se comunica perfeitamente entre si de um modo inteligente. Daqui em diante, os objetos ao nosso redor tendem a "sentir"cientificamente o nosso ambiente, e comunicam entre si de modo a criar um ambiente melhor para nós [6]. Estão assim criadas as condições para um ambiente onde os objetos do dia-a-dia atuam com base no que precisamos e gostamos, sem instruções explícitas.

A Internet das Coisas (IoT) é uma parte integrante da Internet, pode ser definida como uma infraestrutura de rede global dinâmica com capacidades de auto-configuração. Quando se refere a "coisas"em IoT, pretende-se identificar qualquer tipo de objeto através da instalação de sensores ou outro tipo de sistema digital, tornando-o um comunicador inteligente. Essa troca de informação é geralmente feita através de *Radio Frequency Identification*, Wi-Fi, *Ethernet*, *Bluetooth*, entre outras formas de comunicação populares atualmente, é de referir ainda que essa troca de informação pode ser feita pela Internet ou apenas num sistema local como a casa ou o carro do utilizador, por exemplo.

Os autores em [7] concluem que a Internet liga o homem à rede mundial, enquanto a IoT conecta objetos a objetos ou o homem a objetos, logo, é uma extensão da Internet.

É importante entender a estrutura da IoT através da caracterização dos agentes envolvidos e referir a importância das redes de sensores sem fios. Na sua forma mais simples, a IoT é considerada um rede de elementos físicos habilitados por:

- **Sensores:** para recolher informações;
- **Identificadores:** de modo a identificar a fonte de dados;
- **Software:** para analisar os dados;
- **Internet:** para comunicar e notificar.

A ideia principal da IoT passa por conectar fisicamente qualquer coisa/tudo (por exemplo, sensores, dispositivos, máquinas, pessoas, animais, árvores), e criar processos através da Internet com o objetivo de monitorizar e/ou controlar qualquer coisa. Uma abordagem mais completa é incluir normas de regulamentação e processos. Ao respeitar essas normas, a conexão entre dispositivos, e consequente troca de informação é garantida a interoperabilidade entre sistemas o que permite por sua vez a automatização de processos.

Algumas empresas, como a Cisco [6], referem-se a IoT como a *Internet of Everything* (IoE), com quatro componentes-chave: pessoas, processos, dados e coisas. Neste caso, a Internet de Tudo envolve:

- **Pessoas:** Ao conectar as pessoas de várias formas;
- **Dados:** Tomada de decisões através do tratamento de dados;
- **Processo:** Envio da informação certa à pessoa ou máquina certa, no momento certo;
- **Coisas:** Dispositivos físicos e objectos ligados à Internet.

2.3 Sistema ciber-físico

Nos primórdios, o computador era essencialmente um dispositivo único criado com o intuito de prestar apoio ao ser humano, e tinha como principais vantagens possibilitar o processamento de grandes quantidades de informação e resolver cálculos, num menor espaço de tempo. A evolução deste dispositivo foi de tal modo, que atualmente podemos afirmar que o computador é omnipresente, na forma de sistema embebido [8]. Diariamente, de algum modo, interagimos com dispositivos, que integram software e hardware em sistemas mecânicos ou elétricos.

Os sistemas ciber-físicos, designados na literatura como *Cyber-Physical Systems* (CPS), são considerados a nova geração de sistemas, a evolução de processos físicos, com o poder computacional capaz de interagir com humanos. É sem dúvida, um grande avanço tecnológico, a expansão dos recursos físicos já existentes, através da computação, comunicação e controlo.

Estes sistemas vieram melhorar muitas áreas, com inúmeras aplicações, o impacto positivo que teve na economia em geral é tremendo [9]. O conceito de sistema ciber-físico é a generalização de sistemas embebidos, que consiste na aglomeração de vários dispositivos de processamento, com a capacidade de comunicar entre eles, e interagir com o mundo físico, através de sensores e atuadores. Com o aparecimento destes sistemas em diversas áreas, foram-se criando vários modelos consoante a aplicação, pois cada aplicação, tem necessidades específicas, a partir daí, os investigadores tem vindo melhorar em vários aspetos, investindo em novos métodos e ferramentas, através da pesquisa constante, o que resulta, na integração dos princípios de conhecimento em engenharia, nas diversas áreas, como por exemplo (redes, controlo, software, interação humana, aprendizagem,

bem como, elétrica, mecânica, química, biomédica e materiais), de modo a desenvolver estes sistemas, assim como a sua tecnologia de suporte [8], [10].

Existe portanto, uma evolução notória, graças à interdisciplinariedade dos investigadores, no entanto, há que respeitar a arquitetura e requisitos básicos destes sistemas.

Espera-se que cada vez mais, que os sistemas ciber-físicos sejam altamente fiáveis e robustos, uma vez que estes, estão cada vez mais presentes e com aplicações de extrema responsabilidade, é o caso, da saúde, e de qualquer outra aplicação que possa por em causa a segurança de pessoas ou infraestruras. Por conseguinte, os engenheiros envolvidos no desenvolvimento destes sistemas, são os principais responsáveis pelo desenvolvimento de sistemas CPS, onde a mais ínfima falha pode ter consequências catastróficas.

2.3.1 Características

Como referido anteriormente, o facto de os sistemas ciber-físicos serem a integração entre elementos computacionais e físicos, muitas vezes, o sistema implementado, fica algo complexo, independentemente da sua área de aplicação, é portanto, importante saber identificar as principais características, o que por sua vez proporciona uma maior compreensão destes sistemas [9], [11].

Um Sistema ciber-físico, tem as seguintes características chave:

- **Ambiente interactivo**: O sistema é muito restrito em termos de ligações ao processo físico. Qualquer mudança neste processo pode resultar em alterações no comportamento de todo o sistema;
- **Componentes distintos**: Este tipo de sistemas é composto por componentes com diferentes características e capacidade computacional. Por exemplo, em grande parte dos casos, os sensores sem fios, que tem poder de computação reduzido, comunicam os valores de leitura, para máquinas poderosas;
- **Comunicação**: Ao contrário dos sistemas embebidos tradicionais, os sistemas ciber-físicos, requerem uma ligação entre componentes.

Posto isto, é importante referir que os sistemas ciber-físicos, são sistemas sócio-técnicos abertos, ou seja, existe sempre uma interação entre as pessoas e a tecnologia [12]. Em suma, existe uma gama de novas funcionalidades, serviços e recursos que vão muito além dos recursos atuais de sistemas embebidos com comportamento controlado . É de referir no entanto, os pontos destes sistemas a ter em conta, como por exemplo, a gestão das interações ciber e física, garantir segurança, eficiência energética, interoperabilidade e sustentabilidade [11]. Nesta tese, é dada uma atenção especial à questão da segurança, como se sabe é um dos aspetos mais importantes.

Estas são as características mais relevantes de modo a conseguir implementar um sistema ciber-físico viável.

Em suma, é ainda interessante ilustrar estes sistemas (figura 2.1), representando os pontos importantes, a verde, assim como requisitos, propriedades e aplicações. Deste modo, é possível ter uma visão mais global destes sistemas.

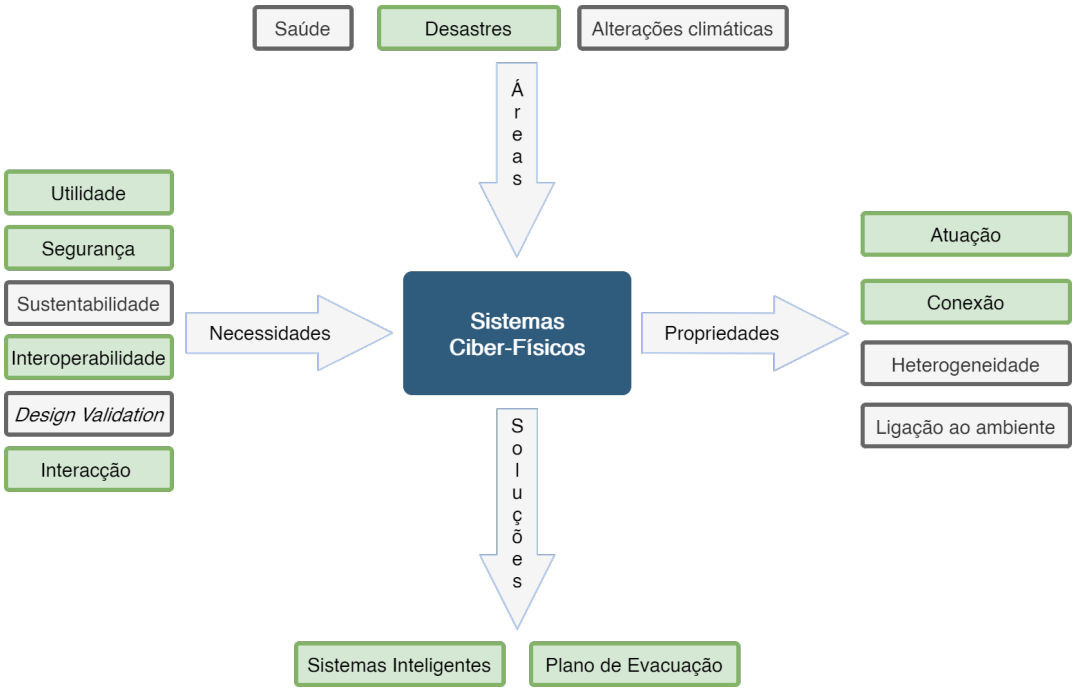

Figura 2.3: Sistema Ciber-Físico em detalhe, adaptado de [11]

2.3.2 Arquitetura

Antes de apresentar em detalhe todas as aplicações possíveis com estes sistemas, é também importante conhecer a sua estrutura. Todos os sistemas ciber-físicos, partilham a mesma a arquitetura, independentemente da sua área de aplicação, diferem depois, na sua abordagem técnica, devido à diversidade de áreas em que estes sistemas estão presentes [13].

Geralmente, os sistemas ciber-físicos são constituídos por 3 camadas:

- **Camada Física**: É a base da estrutura de qualquer sistema ciber-físico, estes sistemas recorrem a dispositivos, como sensores ou atuadores, de modo a interagir diretamente com o mundo físico. Estes dispositivos físicos, estão de algum modo conectados, com ou sem fios. Recorre-se com alguma frequência a tecnologias como, redes móveis(*Segunda Geração*, *Terceira Geração*, *Long Term Evolution*), *Wireless Fidelity* (através das normas *Institute of Electrical and Electronics Engineers 802.11/802.15*), *ZigBee*, *Bluetooth*, *RFID*, entre outras;

- **Camada de Transporte**: Tem um papel chave nos sistemas ciber-físicos, é responsável pelo transporte dos valores lidos pelo sensor. Os dados são encaminhados através da rede ou dentro da rede privada do próprio sistema;

- **Camada de Aplicação**: A terceira e mais interativa de todas as camadas é responsável pelo processamento da informação, sendo esta recolhida através da camada de transporte.

  Como resultado final, os sistemas ciber-físicos deram origem a inúmeras aplicações em diversas áreas, nomeadamente a da saúde, da energia e da indústria, assim como a melhoria de casas e edifícios inteligentes, entre outras inovações. Ainda mais importante, é o facto destes sistemas poderem melhorar a segurança de pessoas e infraestruturas.

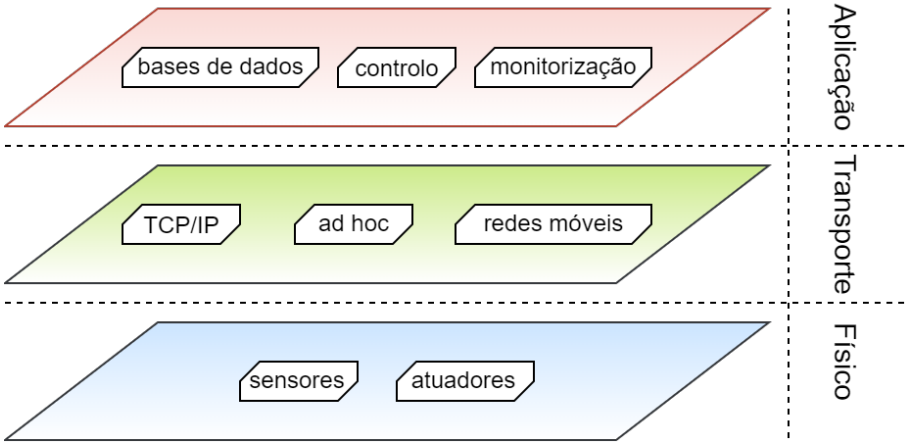

Figura 2.4: Estrutura de um sistema ciber-físico

2.3.3 Aplicações

A investigação e o desenvolvimento dos sistemas ciber-físicos proporcionaram uma notória evolução em diversas áreas, sendo para tal necessário identificar as suas necessidades e desafios. Estes sistemas modernizaram várias áreas através de um leque de novas opções, graças à colaboração de investigadores de diversas áreas e empresas, em virtude da sua multidisplinaridade. Até à data, o investimento realizado em tecnologias de apoio aos sistemas ciber-físicos tem sido significativo, contudo sempre com perspectiva a curto prazo.

No entanto, recentemente, os governos e os diferentes setores da indústria têm vindo a alterar o seu pensamento ao investir a longo prazo. Por exemplo, a União Europeia tem-se destacado através da criação de fundos publico-privados para este fim, nomeadamente, o programa ARTEMIS (Advanced Research and Technology for Embedded Intelligence Systems) [9] .

Anteriormente, na figura 2.3, são ilustradas algumas áreas de aplicação para este trabalho, no entanto, é importante conhecer outras e como os sistemas ciber-físicos podem contribuir, caracterizado na tabela 2.3.

Tabela 2.3: Sistemas ciber-físicos - Outras áreas de aplicação e respetivas características

Áreas de aplicação	Características
Ambiente	São utilizados essencialmente para monitorização, devido a vastas áreas de florestação, montanha e mar, os sistemas ciber-físicos devem operar autonomamente por longos períodos e consumindo o mínimo de energia. A ideia é ter uma rede ad-hoc de sensores ligados a uma máquina poderosa.
Infraestruturas	Os edifícios inteligentes são cada vez mais uma realidade, o papel de um sistema ciber-físico será monitorizar e controlar tudo num edifício.
Saúde	Na área da saúde, os sistemas ciber-físicos podem trazer melhorias, nomeadamente em infraestruturas hospitalares e dispositivos médicos, facilitando a vida a profissionais de saúde e utentes.
Segurança rodoviária	Os sistemas ciber-físicos presentes em automóveis requerem cada vez mais poder de computação, pois é iminente a integração de sistemas de navegação com semáforos e aperfeiçoamento da condução autónoma.

2.4 Sistemas ciber-físicos colaborativos

É importante distinguir IoT de sistema ciber-físico. O conceito de IoT pode ser descrito através de vários cenários em que a ligação à Internet é um pré-requisito obrigatório, assim como as capacidades computacionais aplicadas a um conjunto de objectos e dispositivos. Dessa perspectiva, pode ser difícil distinguir sistemas ciber-físicos de IoT, não fosse o facto de os sistemas ciber-físicos não incluirem necessariamente ligação à Internet. Além disso, os sistemas ciber-físicos têm fronteiras bem claras em termos de separação de elementos internos e externos, enquanto em IoT essa separação não é tão clara. No entanto, os dois conceitos tendem a fundir-se [14].

Ao fazer jus ao próprio nome, sistemas ciber-físicos colaborativos, após a definição individual destas palavras é possível obter uma definição clara e sucinta deste conceito, que não é mais que sistemas ciber-físicos terem a capacidade de colaborar através da troca de informações e de uma eficaz comunicação.

1. **Sistema:**

    a) Um sistema é um conjunto organizado de entidades, que interagem entre si num determinado espaço temporal;

    b) Pode ser considerado "uma entidade no seu todo", com separação bem clara entre elementos externos e internos;

2. **Ciber-físico:** Os sistemas ciber-físicos são a integração perfeita de algoritmos e componentes físicos;

3. **Colaborativo:** A colaboração é um processo em que algumas entidades (organizações, pessoas, dispositivos) trabalham em conjunto para alcançar algumas metas comuns e compatíveis, que dificilmente são alcançáveis sem esforços conjuntos.

Um sistema ciber-físico colaborativo é composto por componentes físicos (*hardware*) e cibernéticos (*software*), privilegia a troca de informação através da comunicação, e atua em conjunto com outras entidades de modo a atingir um objetivo em comum, operando durante um determinado período de tempo [15].

Algumas características destes sistemas colaborativos podem ser a base de estruturas com enorme potencial, em diversas aplicações:

- Organização de comunidades (dispositivos, sistemas e seres humanos);
- Esforços conjuntos;
- Forte trabalho em rede entre componentes;
- Maior nível de inteligência e autonomia.

Uma estrutura de gestão de risco de desastre num edifício, só tem vantagens em ter como base um sistema ciber-físico colaborativo, o objetivo passa por manter tanto a infraestrutura do edifício segura, assim como a sua comunidade, e preparada para incêndios. Uma vez que tudo está ligado, os edifícios são considerados sistemas ciber-físicos, e são parte de uma rede colaborativa destes sistemas.

Seres humanos e dispositivos interagem e cooperam em harmonia, com o objetivo comum de reduzir o número de pessoas afetadas em desastres. Uma proposta de estrutura interessante será uma rede colaborativa, onde um ser humano pode representar um sensor, e um edifício ter a capacidade de supervisão através da implementação de um sistema SCADA.

2.5 Desastres

Segundo um relatório da UNESCO [16] (*United Nations Educational, Scientific and Cultural Organization*), de um modo geral, um desastre é definido como uma perturbação grave no funcionamento de uma sociedade. É potencialmente devastador, pois pode implicar perdas humanas, materiais e económicas, e, na pior das hipóteses, a sociedade afetada não tem meios de resposta a esse desastre. O risco de ocorrer um desastre resulta da combinação de vulnerabilidades com a imprevisibilidade da mãe natureza.

2.5.1 Desastres Naturais

Todas estas perturbações que afetam várias zonas do planeta, são estudadas e previstas, há por isso um trabalho nesta área com o objetivo de preparação e minimização dos estragos. É por isso relevante compreender e identificar cada tipo de desastre de ordem natural e os tipos de riscos que acarretam. Deste modo, é apresentada uma lista com os riscos mais comuns, que são continuamente estudados, previstos e alertados pela Organização Meteorológica Mundial [17]:

- **Meterológicos**: furacões, tornados, raios, incêndios;
- **Hidrológicos**: inundações, *tsunamis*;
- **Geológicos**: vulcões, terremotos, deslizamento de terras;
- **Astrofísicos**: queda de meteoritos;
- **Biológicos**: epidemias, pragas;
- **Antrópicos**: conflitos armados, terrorismo, poluição, incêndios, colapso de infraestuturas;
- **Alterações climáticas**: aumento da frequência e da intensidade das tempestades, subida do nível do mar com consequência do derretimento dos glaciares.

Com o intuito de entender o impacto destes fenómenos na sociedade foi realizado um inquérito, a maior fatia dos participantes pertence à comunidade universitária. Pretende-se identificar os desastres naturais mais temidos pela população, bem como o comportamento que a mesma adota numa situação de emergência. O objetivo deste inquérito é identificar as principais necessidades e analisar o comportamento dos inquiridos, tentando encontrar as melhores soluções tecnológicas. Assim, o inquérito esteve disponível durante 14 dias através de plataformas sociais.

A amostra é composta por 216 participantes, sendo que 60,45 % são do género feminino e o 39,55% do género masculino. Relativamente ao escalão etário, cerca de 64% dos inquiridos tem a idade compreendida entre os 21 e os 30.

Quando inquiridos acerca dos desastres que mais os intimidam, as opções mais selecionadas pelos participantes foram "*Tsunami*", "Incêndio" e "Terramoto", tal como se verifica no gráfico seguinte.

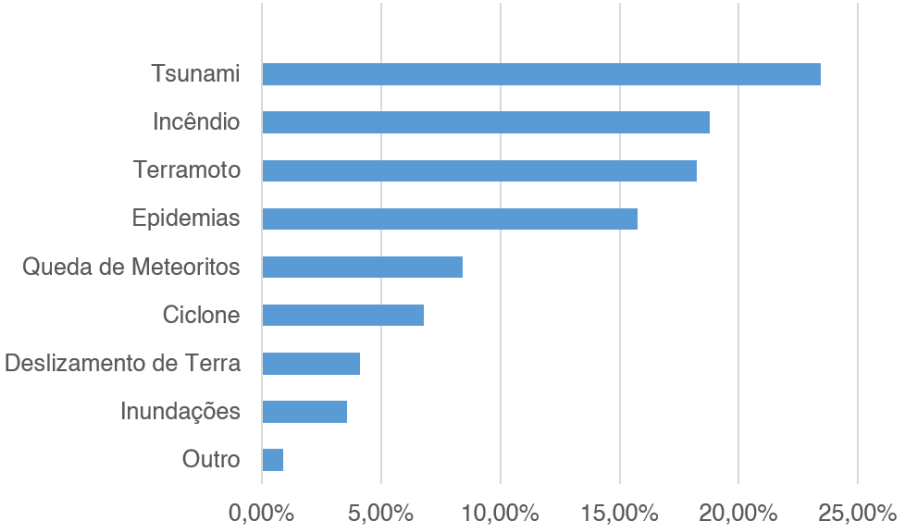

Figura 2.5: Desastres naturais mais temidos

Dos inquiridos que já presenciaram um desastre, cerca de 65% dos mesmos afirmou não existir um plano de evacuação. Relativamente ao sentimento de segurança transmitido pelos planos de evacuação que existem atualmente, mais de metade dos inquiridos não confia nos mesmos.

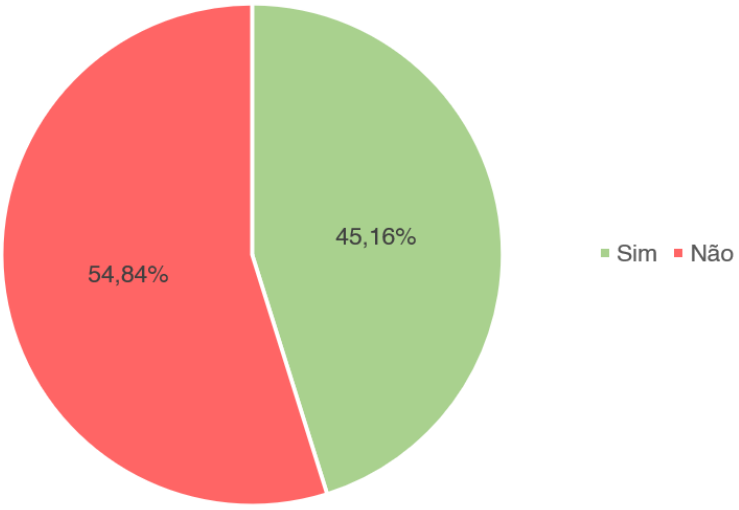

Figura 2.6: Sentimento de segurança

Outra das questões realizadas tinha o objetivo de entender a reação dos inquiridos na seguinte situação: "Está no 9º andar de um edifício e, de repente, soa o alarme de incêndio, e esse incêndio está realmente a acontecer. Ordene as seguintes ações por ordem de prioridade, sendo a 1 mais prioritária e a 5 menos prioritária. ". Através da análise do gráfico seguinte é possível concluir que interpretar o plano de evacuação existente é o fator mais prioritário para mais de 60% dos inquiridos. Em contraste, ligar a um familiar é tida como a opção menos prioritária por cerca de 48% dos inquiridos.

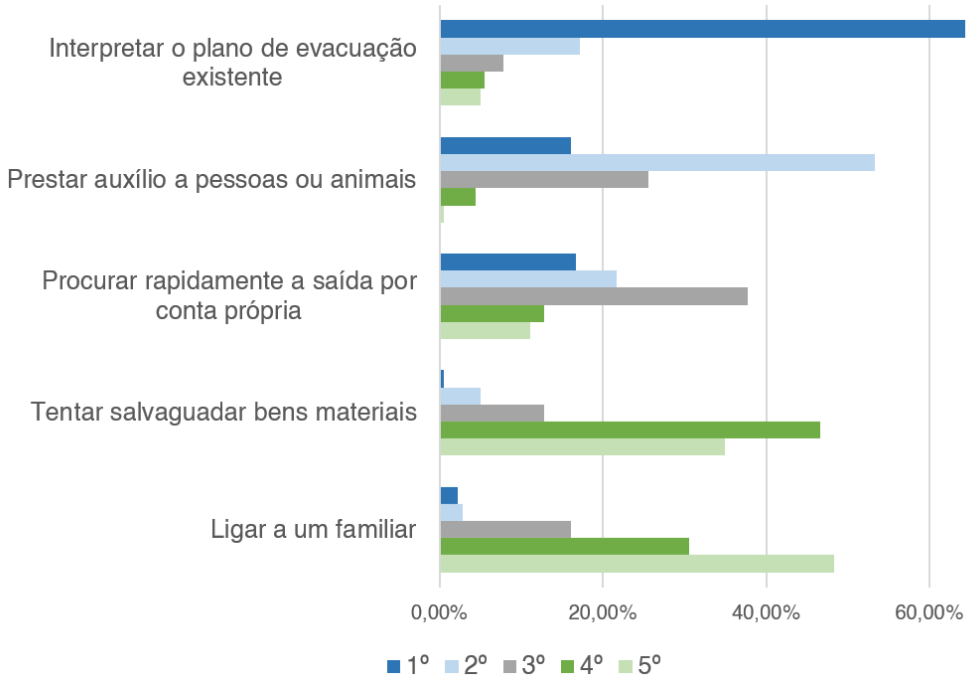

Figura 2.7: Lista de prioridades

Por fim, foi questionado até que ponto os inquiridos confiavam num novo modelo de plano de evacuação inteligente controlado por um computador, que tivesse a função de gerar um caminho seguro até à saída. Assim, cerca de 84% dos participantes confiava nesse modelo, sendo a faixa etária dos 21 aos 30 anos que se demonstra mais recetiva a um sistema deste género. No entanto, surpreendentemente, para a faixa etária acima dos 50 anos, apenas um inquirido respondeu não confiar neste tipo de modelo.

2.5.2 Ataques Cibernéticos

Posto esta lista, é de referir também os ataques cibernéticos, que também estes são um risco da ocorrência de desastre, através de ataques cada vez mais sofisticados e que abrangem diversas áreas. O que torna as infraestruturas mais vulneráveis a ataques é o facto de todas as áreas, sem exceção, estarem cada vez mais dependentes de sistemas computacionais, já não há divisão entre o mundo digital e os processos físicos, ou seja, os sistemas computacionais controlam os processos físicos que outrora eram controlados diretamente e apenas por humanos [18].
O crime cibernético prejudicou mundialmente em aproximadamente 900 mil milhões de euros, ou seja três vezes mais que o dano causado pelos desastres naturais (300 mil milhões de Euros) em 2017, de acordo com o relatório Cyber Handbook 2019 [19] da Marsh & McLennan Companies. A maioria dos ataques cibernéticos que afetam empresas privadas e entidades públicas, geralmente provocam danos económicos e psicológicos, no entanto com o incrível aumento e sofisticação destes ataques, é apenas uma questão de tempo até um ataque cibernético ter consequências catastróficas.

É de referir alguns ataques que podiam ter consequências mais graves [18], em 2017, foi criado o vírus *WannaCry*, que afetou várias empresas e países, com intuito de "sequestro"de dados, pedindo o resgate em *Bitcoin*, posteriormente, em 2018, infetou 200 mil computadores do Sistema Nacional de Saúde do Reino Unido, ainda em 2017, o vírus *NotPetya* paralisou entidades públicas, bancos e agências governamentais na Ucrânia, afetou também a rede de fornecimento energético ucraniana, espalhando-se depois pelo mundo inteiro, em 2018, houve registo de ataques nos portos marítimos de Barcelona e San Diego.

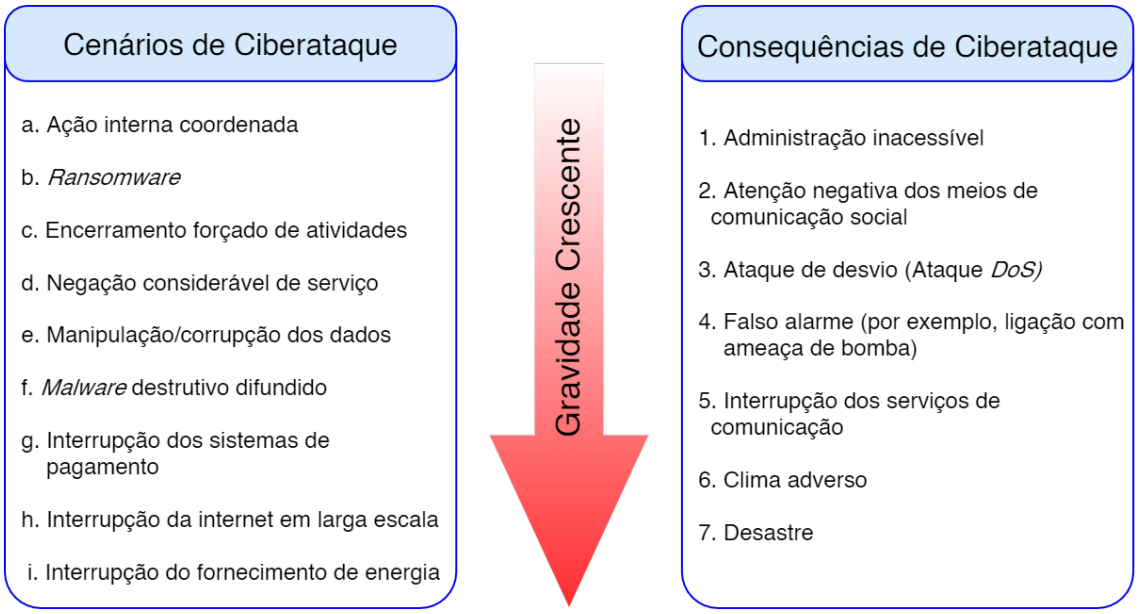

Figura 2.8: Cenários de Ciberataque, adaptado de [19]

2.5.3 Ciclo de Gestão de Risco de Desastre

Posto isto, pode-se concluir que os ataques cibernéticos partilham os mesmos princípios de gestão de desastre, como se de um desastre natural se tratasse.

A gestão de risco de desastre (GRD) é geralmente representada como um ciclo, sendo possível identificar facilmente três estágios, nomeadamente, antes, durante e após o desastre, em cada estágio existem atividades, medidas e reacções [16].

Antes da ocorrência de qualquer evento catastrófico é de extrema importância fazer uma avaliação do risco e a estruturação de medidas preventivas, de modo a minimizar o impacto de um eventual desastre. É ainda essencial a criação de um plano de emergência e procedimentos de evacuação, assim como possuir um sistema de alarme eficaz. Na momento do desastre é essencial seguir todos os procedimentos de emergência, onde todos os detalhes contam para salvar pessoas. Após o desastre ter ocorrido é tempo de reflexão e análise. É o momento onde são avaliados os danos causados, de modo a proceder à recuperação de um modo gradual.

O ciclo de gestão de risco é uma ferramenta eficaz, no entanto, a comunicação e o acompanhamento ao longo dos estágios são considerações essenciais ao seu bom funcionamento [16].

Figura 2.9: Ciclo GRD, adaptado de [20]

2.5.4 Estrutura *Sendai*

Antes de abordar os planos de evacuação é importante referir a estrutura *Sendai*, esta estrutura tem como objetivo a redução do risco de catástrofe, para 2015-2030, foi o primeiro grande acordo de desenvolvimento pós 2015, e proporciona aos estados membros acções concretas para manter o desenvolvimento sustentável e proteger os países do risco de desastre [21]. Os objetivos e metas que esta estrutura pretende alcançar são em parte os desafios a que os sistemas ciber-físicos se propõem, pode-se assim relacionar naturalmente a estrutura *Sendai* com os sistemas ciber-físicos. Esta estrutura pretende reduzir o risco de desastre através do estabelecimento de 7 objetivos globais, 4 destes objectivos focam-se em reduzir as perdas e os danos causados por desastres, e os restantes 3 pretendem assegurar a sua eficácia através de processos para tal.

Até ao ano de 2030, esta estrutura tem como objetivo reduzir substancialmente:

- A mortalidade por desastre;
- O número de pessoas afetadas em desastres;
- Perdas económicas causadas por desastres em relação ao PIB (Produto Interno Bruto) global;

- O dano causado por eventos catastróficos em infraestruturas e em serviços básicos.

Por outro lado, a estrutura *Sendai* pretende aumentar de um modo gradual:

- O número de países com estratégias de redução de risco de desastre;
- A cooperação internacional com os países em desenvolvimento, através de um apoio adequado e sustentável;
- A disponibilidade a sistemas de alarme eficazes, assim como o acesso a informações e avaliações de risco de desastre para pessoas.

Além dos desafios em comum desta estrutura com os sistemas ciber-físicos, é de notar também o facto das áreas de aplicação serem as mesmas. A estrutura *Sendai* é aplicada em áreas como saúde e bem estar, protecção do ambiente, transportes, comunicação, infraestruturas em geral e rede de distribuição de energia elétrica, estas são apenas a áreas em comum com os sistemas ciber-físicos.
Segundo o relatório anual referente ao ano de 2019 [21], foi possível verificar que 81 estados membros já possuem um estratégia de redução de risco de desastre com base na estrutura *Sendai*, e que existem atualmente cerca de 4311 cidades com campanhas de sensibilização de resiliência em desastres, foram ainda contabilizados 4087 funcionários dos governos e partes interessadas na redução de risco de desastre com formação por parte das Nações Unidas, 39 por cento dos quais eram mulheres. Em 2019, a estratégia de redução de risco de desastre das Nações Unidas reforçou ainda mais a sua eficiência operacional ao recorrer à tecnologia para melhorar a gestão baseada em resultados, planeamento, monitorização e avaliação. Esta estratégia conta também com uma gestão de recursos humanos eficaz através da formação do seu pessoal.

Esta é a política das Nações Unidas para melhorar o desempenho operacional em situações de desastre.

2.5.5 Sumário

Devido ao desenvolvimento económico e consequente crescimento urbano, os desastres têm tido um maior impacto. No entanto, atualmente, graças à evolução tecnológica, é possível prever certos desastres de ordem natural, como por exemplo, recorrer a imagens detalhadas de satélite que nos permite identificar e acompanhar todos os estados de um ciclone, ou alertar para a formação de *tsunamis* através de bóias estrategicamente distribuídas pelos oceanos. Contudo, alguns riscos de desastres são resultado da atividade humana, nomeadamente, a desflorestação, construção de estruturas sem o devido cumprimento das normas de engenharia, produção de energia nuclear ou até mesmo um ataque cibernético.

Apesar de distintos, estes acontecimentos têm em comum o risco inerente do ser humano estar exposto a danos físicos ou psicológicos, direta ou indiretamente. Desta

forma é importante ter uma evolução balanceada da tecnologia no aspeto da segurança, podendo só assim cumprir os objetivos da estrutura *Sendai*.

2.6 Planos de evacuação

Os planos de evacuação, especificamente em edifícios, não são mais que uma estratégia de saída. Tradicionalmente incluem abordagens de evacuação, realocação ou proteção no local que podem ser combinadas ou faseadas, dependendo das metas e objetivos gerais, como se pode verificar no anexo A. Os planos de evacuação de um edifício podem incluir a evacuação simultânea total ou parcial dos ocupantes do edifício durante situações de emergência. Para estratégias de protecção no local, pode não existir evacuação, o plano passa por sensibilizar as pessoas a ficarem onde estão. As estratégias de evacuação por fases recorrem às sinalizações de emergência de um modo eficiente. Os planos de evacuação devem considerar os utilizadores de cadeiras de rodas e outras pessoas com algum tipo de deficiência e mobilidade reduzida. Consideremos uma estratégia de evacuação típica de um arranha-céus, que exige a evacuação dos ocupantes de três andares durante um cenário de incêndio - o andar de alarme, um andar acima e um andar abaixo. Se tudo correr bem, e se o plano for bem sucedido através de vários sistemas ativos e passivos, o fogo é controlado ou suprimido pelos sistemas apropriados, os ocupantes de outros pisos podem permanecer no local durante algum tempo e podem nem sequer precisar de ser alertados para a situação de incêndio. Neste cenário, os ocupantes de outros andares seriam orientados a permanecer no local, ou não seriam sequer notificados da situação de incêndio.

O esquema de evacuação por fases permite também evacuar ocupantes de outros andares, dependendo do progresso das atividades e da supressão do incêndio, ou verificar o estado do alarme. Os edifícios altos mais modernos também podem incorporar elevadores especialmente protegidos e controlados para evacuar os ocupantes dos andares superiores de forma eficiente. Podem ser usadas estratégias de realocação, de modo a guiar os ocupantes para longe das proximidades do perigo de incêndio e para um local mais seguro.

2.6.1 SCADA

A Internet das Coisas (IoT) está a delinear o setor das tecnologias de informação e comunicação. A possibilidade de combinar o mundo real e virtual através da implementação massiva de dispositivos incorporados permite novas oportunidades. Este processo de digitalização é a base da evolução na área de sensoriamento inteligente, proporcionando um poder acrescido de monitorização e controlo, tanto em edifícios inteligentes, como em processos industriais.

Um edifício inteligente, que tenha um sistema SCADA, pode ser um auxílio fulcral na ocorrência de desastres, pois fornece dados precisos às autoridades de resgate.

Para além disso, pode fazer parte de uma estrutura que através desses dados faça parte de um plano de evacuação ou mobilize as pessoas para um local seguro, enquanto as autoridades de resgate não chegam.

Um sistema SCADA tem como principais trunfos a supervisão e aquisição de dados. Tipicamente, é composto por vários elementos, podendo estes ser *hardware* ou *software*. É importante referir a HMI (*Human-Machine Interface*), que é um elemento muito importante destes sistemas, pois permite os operadores analisarem toda a integridade do edifício, como por exemplo câmaras de videovigilância, sensores, sistemas de ventilação em apenas um monitor [22].

Ao utilizar os sistemas SCADA, as empresas ou entidades podem monitorizar todo o edifício abrangido pelo sistema de forma local ou remota, e interagir diretamente com equipamentos individuais, como sistemas Aquecimento, Ventilação e Ar Condicionado, alarmes e elevadores. Em alguns casos, estes sistemas podem controlar automaticamente alguns equipamentos com base nos dados recebidos. Os sistemas SCADA permitem ainda a elaboração de relatórios completos do edifício com base em dados em tempo real e arquivam os dados para uma análise posterior .

2.6.2 Perspectivas

Os algoritmos e protocolos de encaminhamento podem ter um papel essencial na otimização dos planos de evacuação, uma vez que têm a grande vantagem de se adaptarem a diferentes cenários de desastre em redes distribuídas. No entanto, a melhor forma de abordar a evacuação de pessoas e o comportamento de sistemas é, sem dúvida, através de modelos matemáticos, pois não basta a criação do caminho mais curto e seguro, existem outros fatores importantes como a formação de multidões.

Proporcionar o caminho mais curto tendo em conta o comportamento das pessoas, ou melhorar a comunicação numa situação de desastre, pode ser a chave para uma evacuação bem sucedida. Perspetiva-se num futuro próximo o forte contributo das tecnologias que dê origem a soluções, onde a principal meta será sempre minimizar o impacto do desastre.

O padrão psicológico de evacuação e outros fatores como a formação de multidões, geralmente, não é tido em conta numa estrutura de gestão de risco de desastre. Os estudos acerca do planeamento dinâmico de evacuação são a base para o processo de tomada de decisão em tempo real como resposta às situações de emergência [23].

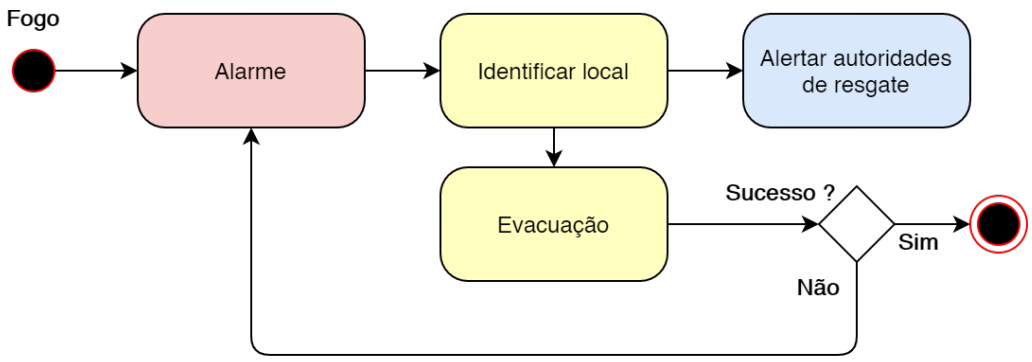

Figura 2.10: Fluxo de plano de emergência

Nos sistemas tradicionais de emergência, os sensores alertam para a anomalia soando o alarme, com o intuito de alertas as pessoas que se encontram no edifício assim com as autoridades. Posteriormente os bombeiros deslocam-se o mais rápido possível para o edifício afetado, de modo a extinguir o fogo e prestar primeiros socorros se assim for necessário, e enquanto isso, as pessoas que se encontram dentro do edifício nesse momento devem de um modo calmo seguir o plano de evacuação. A figura 2.10 permite observar a sequência de acontecimentos num caso deste género.

Com tecnologia cada vez mais presente nos planos de evacuação é possível obter planos mais eficazes. Seguindo sempre este modelo base é possível que surjam otimizações do mesmo com a possível adição novos módulos dependentes apenas de tecnologia.

CCRP

Do trabalho já existente, o CCRP, *Capacity Constrained Route Planner*, é um planeador de evacuação otimizado que permite reduzir o tempo de execução do planeamento de evacuação ideal. O CCRP, assim como as suas otimizacões, são baseadas no algoritmo de *Dijkstra*. Idealmente o planeador de evacuação requer um gráfico de tempo expandido e, portanto, como consequência requer mais memória e o tempo de execução será mais longo em relação ao CCRP tradicional, no entanto o CCRP não requer um gráfico com tempo expandido, pois este constrói um plano de evacuação baseado no *Earliest Arrival time*, (EA) a partir dos nós de origem.

Este planeador encontra os caminhos mais curtos de todos os nós de origem para todos os destinos com base na duração da rota, recorrendo ao algoritmo Dijkstra. Posteriormente, as pessoas em perigo são orientadas desde o seu nó de origem através de uma rota, cujo EA é mínimo, até um nó de destino do caminho, chegando assim à saída ou um local seguro [24].

Apesar do CCRP admitir o máximo número de pessoas possível nas rotas criadas, existe o fator da capacidade, obviamente de edifício para edifício varia a infraestrutura, e como o comprimento e largura de corredores, assim como a dimensão do edifício está diretamente relacionada com o número de pessoas a serem evacuadas, onde o ideal é ter um fluxo equilibrado de pessoas evacuadas, nunca esquecendo a possível formação de

multidões.

Algorithm 1 CCRP

Input:

1. Rede de sensores $G = (N, E)$ onde N é um conjunto de nós e E representa as suas ligações;
2. S: Conjunto de nós origem, $S \subseteq N$;
3. D: Conjunto de nós destino, $D \subset N$;

Output: Plano de evacuação: Criação de rotas

1: Adiciona nó origem S_0
2: Faz a ligação de S_0 a todos os outros nós origem com duração de 0 e capacidade ∞
3: **while** cada nó origem tem pessoas **do**
4: Acha rota R que tenha o melhor tempo EA através do algoritmo Dijkstra
5: Calcula o número máximo de pessoas, f_{max}, que pode ir por R
6: Aloca capacidade f_{max} para o nó e para a ligação
7: **fim while**
8: É gerado o plano de evacuação

O algoritmo começa com a adição do nó origem e faz as respetivas ligações de acordo com a capacidade apropriada. (linhas 1 e 2 do algoritmo 1). Entretanto, o algoritmo através de iterações mantém a rota (*loop* das linhas 3-7) enquanto houver pessoas no nó origem.
Cada iteração começa com a descoberta da rota R, considerando as capacidades disponíveis. Depois, determina o número máximo de pessoas f_{max} suportado pela rota R. Finalmente, a rota R é reservada para f_{max}. O algoritmo só termina quando todas pessoas se encontrarem em segurança através de um rota [25].

2.6.3 Sumário

Uma abordagem a uma situação de emergência tem com base a experiência de acontecimentos passados, o que permite agilizar os processos de evacuação através da seleção de cenários pré-planeados. No entanto, os desfechos dependem sempre de informações precisas em tempo real fornecidas às autoridades. Assim, é necessário que estas tomem decisões eficazes, reajam de acordo com a situação e transmitam as instruções adequadas. Deste modo, o pré-planeamento e os simulacros são essenciais. A estratégia de evacuação irá depender sempre da estrutura e características do edifício, tais como portas, corredores, escadas e saídas, de modo a facilitar a evacuação segura.

A abordagem destas tecnologias visa melhorar todo o processo de evacuação respeitando sempre os padrões tradicionais de segurança, e não pretende necessariamente, substituir os atuais planos de evacuação. É apenas desenvolvido como complemento do mesmo através de inovação, e é fruto da evolução tecnológica.

Capítulo 3

Caracterização da Estrutura

A rede de sensores IoT deve ser versátil, adaptável a diferentes cenários, mais concretamente redes que variam a sua topologia frequentemente. Uma potencial estrutura de gestão de risco de desastre da nova geração deve seguir os pontos de desenvolvimento desta tese:

1. Implementação e simulação da rede de sensores IoT num caso prático;

2. Proceder à manipulação de valores de temperatura e frequência de envio de informação. E assim, analisar e comparar a rede em condições normais e com atividade fora do comum, para efeitos de simulação.

3.1 Planeamento

O grande desafio desta tese passa por otimizar os sistemas ciber-físicos, de modo a melhor preparar os edifícios e a população para eventos catastróficos. Uma vez realizada a pesquisa ao longo da preparação da dissertação, é importante reter conhecimentos de redes de sensores sem fios.

O processo de otimização consiste no ajuste de parâmetros sem violar as normas de arquitetura ou regulamentação. Assim, será analisada e testada uma rede de sensores sem fios, bem como a comunicação entre estes. Geralmente, os objectivos passam por minimizar os custos e maximizar a eficiência da rede. Quanto ao protocolo de comunicação, pretende-se avaliar a sua eficiência, latência, precisão, tolerância a falhas, escalabilidade e exposição dos sensores.

Numa fase mais adiantada, pretende-se com base na planta de um edifício e a interpretação do comportamento da rede sensores gerar um plano de evacuação. A rede de

sensores é implementada no simulador *cooja*. Dentro das ferramentas *open source* existentes, esta é a mais apropriada para redes de sensores sem fios, integra o sistema Contiki OS e permite simular e estudar em detalhe especificamente este tipo de redes.

3.1.1 Ambiente

Existe a necessidade de tornar os edifícios num local mais seguro, esta tese tem como principal foco os planos de evacuação, de onde se conclui que a otimização de um plano de evacuação depende do ambiente de um edifício, isto é, um novo modelo de plano de evacuação depende de toda a envolvente do edifício. Neste subcapítulo serão abordados os *smart buildings* assim como a comunicação de uma rede de sensores IoT presente nestes edifícios.

Smart Building

Os *Smart Buildings* são edifícios preparados para integrar conexões sem fios com sensores IoT, de forma a registar e analisar grandes quantidades de dados. Tipicamente, estes sistemas são instalados de um modo independente a outras ligações do edifício, proporcionando uma maior interoperabilidade e independência entre dados e sistema. Estes edifícios inteligentes pretendem garantir a melhor gestão dos recursos, como água e energia, a redução dos custos com operação e manutenção dos edifícios, e a qualidade de vida dos ocupantes.

Os autores em [26] afirmam que o principal objetivo de um *smart building* é conectar dados, pessoas e sistemas . Através desta integração é possível optimizar os sistemas ciberfísicos em desastres, e assim enfrentar os desafios desta tese. Os edifícios inteligentes proporcionam além de uma maior segurança, um serviço eficaz e cómodo para os seus ocupantes.

Comunicação

De todas as formas de comunicação esta tese pretende abordar protocolos de comunicação apropriados a redes instáveis, redes que podem sofrer alterações constantes na sua topologia. Este tipo de redes é composta por muitos dispositivos IoT com energia e memória limitada, no entanto possuem processamento individual, ainda assim com algumas limitações.

Em [27] é sugerida uma arquitectura de comunicação onde um *smart building* recorre a uma *gateway* ou uma solução *cloud* para comunicar com sistemas de automação presente neste tipo de edifícios. A solução *cloud* permite uma maior versatilidade na canalização e tratamento de dados. O tratamento de dados é realizado na *cloud* inclui estatísticas, integração com outros edifícios inteligentes e conversão de protocolos, como por exemplo BACnet.

As redes instáveis e sujeitas a perdas estão presentes em vários cenários, dos quais, automação e monitorização em edifícios, casas inteligentes, área da saúde, monitorização

do meio ambiente, entre outras. Existe particular interesse nos incêndios em edifícios altos para fins de conclusão desta tese.

Os dispositivos inteligentes presentes no edifício comunicam através de IEEE 802.15.4 ou Wi-Fi de baixa potência.

No início, a aplicação de redes de sensores IoT em diferentes cenários sujeitos a perda de dados requeria uma solução de encaminhamento *standard*. De modo a responder às necessidades e requisitos surgiu a criação de *"Ripple"* ou RPL (*Routing Protocol for Low Power and Lossy Networks*). Este protocolo de encaminhamento proativo recorre a endereços IPv6 (*Sexta versão de Internet Protocol*) de modo a construir rotas em intervalos aleatórios, criando ao longo do tempo a topologia da rede, neste caso especifíco, 6LoWPAN, (*IPv6 over Low power Wireless Personal Area Networks*). Sendo do tipo *distance vector* suporta várias rotas de tráfego através de grafos acíclicos direccionados (DAG) para criar um ou mais destinos orientados (DODAGs), que por sua vez podem estar associados a uma ou mais instâncias RPL [28], em 4.2 a descrição da topologia da rede é explicada mais aprofundadamente.

A solução 6LoWPAN+RPL é portanto adequada a redes com estas características, particularmente a sensores operados por bateria que desligam o seu módulo rádio durante longos períodos de tempo, não mantendo a sincronização - fenómeno de *duty cycle*.
Por outro lado existem protocolos reativos que abordam este tipo de redes de um modo diferente. O AODV, *Ad-hoc On Demand Distance Vector*, é um protocolo de encaminhamento reativo, isto é, uma rota só é criada quando um nó origem faz um pedido para enviar pacotes para um nó destino. Uma vez estabelecida, a rota entre ambos é mantida durante o período de duração da comunicação. Existem três tipos de mensagens distintas que permitem estabelecer a rota, assim como a sua manutenção:

- Pedido de Rota (*Route Request*);
- Resposta de Rota (*Route Reply*);
- Erro de Rota (*Route Error*).

Estas mensagens são recebidas através de datagramas (UDP, *User Datagram Protocol*), utilizando um cabeçalho do Protocolo Internet (IP).

Durante o tempo de ligação estipulado, do tipo TTL (*Time To Live*), a rota é gerada através de pacotes RREQ. A mensagem RREQ é enviada a partir de um nó origem, até que o nó destino ou nós vizinhos encontrem a rota pretendida. Posteriormente, o nó destino envia a RREP, resposta à rota, esta percorre a rota inversa até ao nó origem, que verifica o número sequencial de destino (identificador). Se o RREQ for recebido mais do que uma vez, estes serão descartados. Se for detectada uma quebra na ligação, é enviada uma mensagem de erro de rota (RERR) para o nó origem. Uma vez enviado o RERR, cada nó intermediário invalida as rotas para esse nó destino [29].

Discussão

Esta dissertação, para além do conceito de *smart building* pretende realçar a versatilidade dos dispositivos IoT. O facto de qualquer tipo de edifício independentemente da época se poder transformar num *smart building* é fascinante.

Para tal, não basta espalhar uma série de sensores por um edifício obsoleto, esta interoperabilidade só é possível alcançar recorrendo a outras tecnologias, nomeadamente a tecnologia *cloud* e a iminência do 5G (Quinta Geração), além da questão energética, naturalmente.

Em relação à comunicação entre dispositivos, existem opções variadas, desde logo surgem algumas vantagens do AODV em relação a outras opções [30], como, o facto de ser económico quanto à criação de rotas, criando apenas quando necessário, a poupança da largura de banda e energia durante a sua inatividade, e a sua capacidade de interligar até milhares de nós móveis fazem deste protocolo uma boa opção. No entanto existem algumas desvantagens como alguma falta de segurança, a alta latência no encaminhamento e determinados casos em que os nós intermédios podem criar rotas mais atualizadas, que ainda assim podem ser obsoletas, podendo dar origem assim a um problema de escalabilidade. É ainda relevante abordar o LOADng, (*Lighweight On-demand Ad hoc Distance-vector Routing Protocol – Next Generation*), uma versão simplificada do AODV. Apesar dos mesmos princípios de funcionamento, este protocolo de encaminhamento preza pela simplicidade e recurso a pouca memória.

Os protocolos de encaminhamento mencionados neste capítulo são do tipo reativo, geralmente são adequados para topologias de rede de grandes dimensões, uma vez que "inundam"a rede com mensagens de controlo RREQ para obterem o melhor caminho para o destino. Este processo pode dar origem a alguma latência, pois os nós têm de calcular sempre a rota antes de qualquer envio. Os nós da rede não necessitam de estar constantemente a trocar informação caso não existam pacotes a serem encaminhados. As rotas são apenas estabelecidas quando é necessário.

O protocolo de encaminhamento escolhido para esta simulação foi o RPL. Todos os protocolos de encaminhamento têm vantagens e desvantagens, para efeitos de simulação para além de conveniente, o RPL mostrou ser o mais adequado a este caso prático. A fácil integração com o simulador *cooja* foi também um fator determinante na escolha. Além do plano de simulação, estes sensores são reais, o que torna esta simulação muito próxima da realidade. A implementação de outros protocolos necessitava de adaptação a este simulador, é preciso salientar que todo o algoritmo que permite a comunicação entre sensores já vem incluído no simulador.

Em 4.2, é possível entender o funcionamento do RPL, um padrão em IoT. Apesar de não ser o foco desta tese, terá um papel importante.

Seguidamente, na secção 3.1.2, é feita uma descrição das ferramentas utilizadas que tornam esta tese possível, além do protocolo de encaminhamento, as restantes ferramentas foram escolhidas de um modo ponderado a pensar na integração com outras tecnologias sem alterações significativas, prezando assim pela interoperabilidade.

3.1.2 Ferramentas

No tema das redes de sensores sem fios é importante ter as ferramentas apropriadas que permitam fazer a simulação de uma rede, assim como uma análise profunda da mesma. Nos dias que correm, com o acesso a apenas uma ferramenta é possível simular uma rede de sensores sem fios sem grandes esforços de programação e *debug*. Uma das opções possíveis passa por comprar um software, algumas empresas vendem os chamados "sistemas operativos em tempo real". Estes sistemas são no entanto direccionados para hardware com muito mais recursos do que os típicos sensores IoT.
Felizmente, existe uma alternativa: a comunidade de investigadores de redes de sensores sem fios deu resposta à necessidade de um sistema operativo facilmente adaptável, o que permite a abordagem a vários tipos diferentes de redes de sensores. Com uma abordagem "*open source*"existe uma maior adesão dos investigadores neste tipo de redes, mesmo sem bases de programação, e também só assim é possível manter estes sistemas operativos bastante específicos, e assegurar a sua constante evolução.

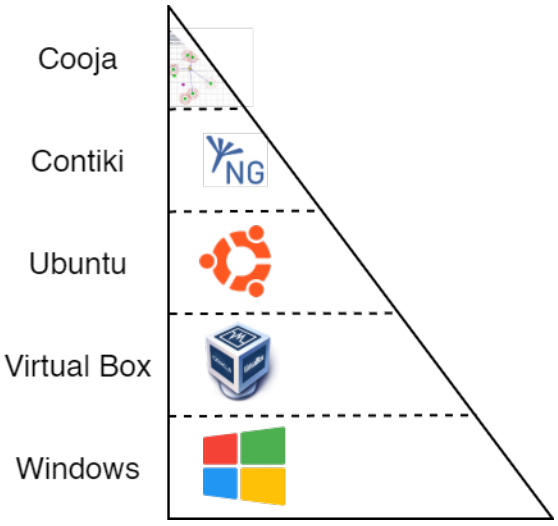

Figura 3.1: Hierarquia de ferramentas utilizadas nesta dissertação

Para a simulação de uma rede de sensores, as ferramentas assentam numa topologia de hierarquia. O sistema operativo *Microsoft Windows 10* é a base onde é realizada toda a pesquisa, escrita, desenho e simulação referentes a esta dissertação. Para poder fazer a simulação é forçoso ter o software *virtual box* de modo a ter a máquina virtual do Contiki OS, outra software que permita correr máquinas virtuais também funcionava. Por sua vez o sistema operativo que permite fazer a simulação de uma rede IoT tem como base o sistema operativo *Ubuntu*. Uma vez tudo instalado e configurado, e por último, é importante executar o simulador *cooja* e assim ambientar com esta ferramenta.

Contiki OS - Cooja

Efetivamente a escolha do sistema operativo para simular a rede de sensores sem fios foi bastante ponderada, o Contiki OS acabou por ser o escolhido, principalmente devido ao facto de ser *open source* e à sua capacidade de análise e teste. Este sistema possui um suporte total para os nós *Tmote Sky* onde é possível simular diferentes cenários integrados através deste tipo de sensor. O sistema operativo é escrito em C e inclui tudo o que é necessário para criar uma rede de sensores IoT que comunicam por RPL e 6LoWPAN.

Em [31] é referido que através da *stack* µIPv6 este sistema suporta ICMP (*Internet Control Message Protocol*), UDP e TCP (*Transmission Control Protocol*), e destaca-se pela sua leveza, com baixo consumo de memória. O Contiki é um sistema flexível a nível de construção e configuração. A variedade de opções deve-se ao facto dos processos serem executados através da linguagem C, podendo inclusive fazer alterações em qualquer altura, no `Makefile` ou no ficheiro de configuração do projecto.

Basicamente, o Contiki é uma enorme colecção de macros escritas em C e um conjunto de `makefiles`, depois de compiladas e ligadas, podem ser utilizadas para efeitos de simulação no *cooja* através de um ficheiro de objectos (sensores mote). As versões sucessivas de todo o código fonte estão disponíveis no *Github*.

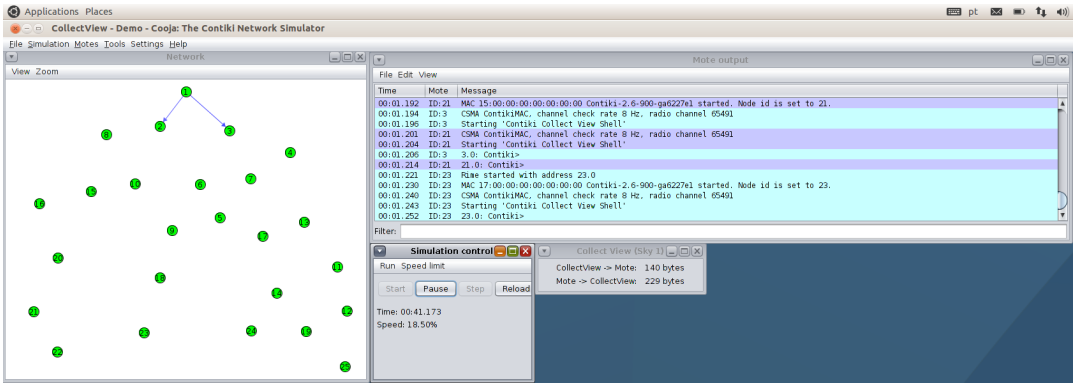

Figura 3.2: Ambiente de trabalho Cooja - Contiki OS

Tmote Sky

Segundo [32], este tipo de dispositivos podem comunicar através de um módulo sem fios de baixa potência, o que é indicado para utilização em redes de sensores sem fios com o objetivo de monitorização. Este modelo é alimentado por duas pilhas AA, pode também receber energia através da sua porta USB (*Universal Serial Bus*). A utilização de baixa energia do *Tmote Sky* deve-se principalmente ao facto da eficiência do seu microcontrolador MSP430 F1611 com apenas 10 *Kilobytes* RAM (*Random Acces Memory*) e 48 *Kilobytes flash* ROM (*Read Only Memory*).

O grande trunfo deste sensor neste tipo de redes é a sua capacidade de ligar rapidamente, ou seja, normalmente este dispositivo está em modo *standby* e pode "acordar"em menos de 6μs. O processador de 16 *bits* permite-lhe utilizar menos energia enquanto está activo e em modo de sono, permitindo que o dispositivo esteja operacional durante anos

com um único par de pilhas AA. Ainda em [32], é referido que o *Tmote Sky* está equipado com uma antena integrada com 50m de alcance no interior e 125m em exterior, o módulo rádio é o Chipcon CC2420 que permite uma comunicação sem fios fiável.

Existem três tipos de sensores integrados no *Tmote Sky*, especificamente de Humidade, Temperatura e Luz. A nível de segurança, a ligação entre sensores e *gateway* ou *sink* é encriptada e requer autenticação. Na figura 3.3 é possível ver o aspeto deste dispositivo.

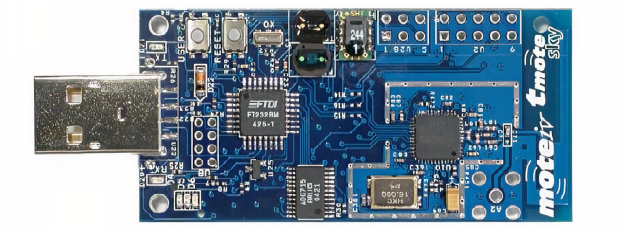

Figura 3.3: Tmote Sky

3.1.3 Estrutura

Esta dissertação aborda essencialmente uma rede de sensores IoT de um caso prático, serão criados processos que permitam o envio de temperaturas entre sensores, o objetivo passa por estudar a rede em diferentes cenários. A estrutura ilustrada na figura 3.4 participou no concurso INCCYBER HUB inovation award 2020. São abordadas as relações entre agentes e são analisados os objetivos e desafios. Por fim, há que depreender a cadeia de acontecimentos dentro do *smart building* e fora deste.

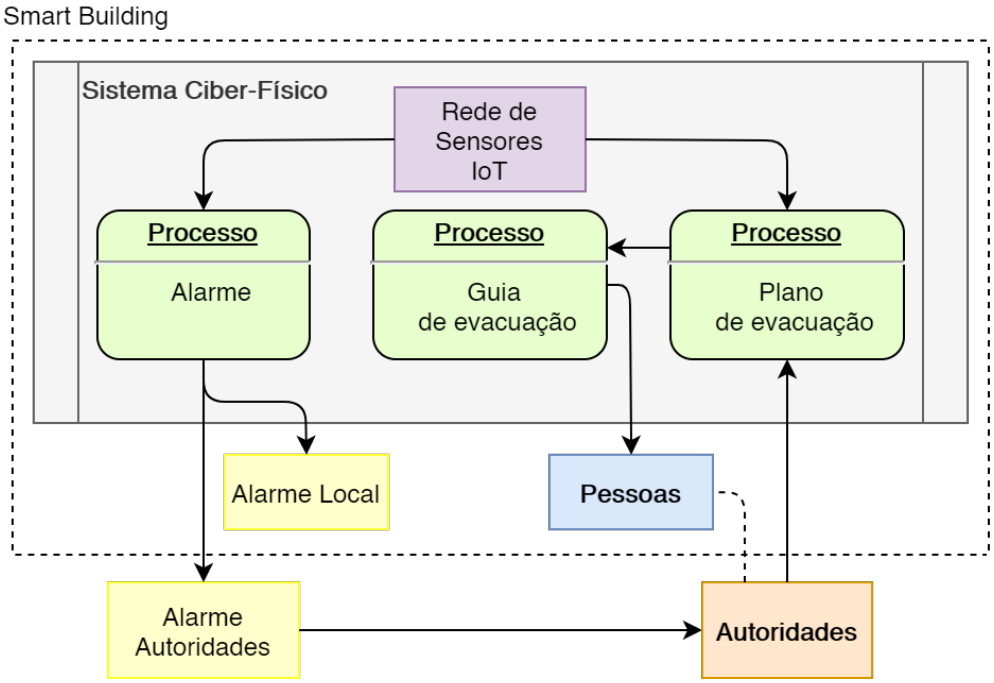

Figura 3.4: Exemplo de Estrutura

Na figura 3.4, é possível observar que o sistema ciber-físico é o conjunto *smart building* e rede de sensores IoT, este sistema dá origem direta e indirectamente aos processos (a verde). A rede de sensores IoT (bloco lilás), é responsável pelos processos alarme e plano de evacuação, com base no plano de evacuação é possível gerar um guia de evacuação, com poder de computação pode-se gerar este guia de acordo com os possíveis cenários de desastre. O plano de evacuação, como tradicionalmente, apoia as autoridade, enquanto o guia de evacuação pretende orientar as pessoas no edifício até estas chegarem ao seu auxílio.

Quanto ao sistema de alarme, para além do alarme local já existente, tipicamente através de uma sirene, existe a possibilidade de simultaneamente as autoridades serem avisadas com informações detalhadas de um desastre.

3.2 Objetivos

O principal objetivo desta dissertação é a aplicação de uma rede de sensores IoT num edifício histórico, e assim demonstrar que é possível tornar qualquer edifício um local mais seguro independentemente da sua infraestrutura.

O Capítulo 3 prepara algum enquadramento complementar e caracterização, este capítulo é essencial a uma melhor compreensão do seguinte capítulo. No início do Capítulo 4 são fornecidos os detalhes da simulação e é apresentada a programação dos sensores, foram configurados dois ficheiros, correspondentes às funções de *sender* e *sink*. A programação dos motes consistiu em acrescentar algumas linhas de código em C, devidamente comentadas. Numa primeira etapa da simulação, os valores de temperatura são gerados através do mote sht11 (*Sensor Humidity Temperature*), representando valores reais. Os motes comunicam entre si com o intuito de trocar os valores das temperaturas nas diferentes divisões, criando assim esta rede inteligente de sensores IoT.

O caso prático consiste na simulação e consequente estudo de uma rede de sensores de um edifício inspirado no *The Plaza*, um hotel construído no início do séc. XX. Pretende-se numa fase mais avançada a simulação de um incêndio, no qual é especificamente programado um ou mais sensores. Posteriormente será feita uma análise, mais especificamente quanto à propagação dessa informação pela rede. Além desta particularidade serão analisadas as típicas características da rede de sensores, nomeadamente propriedades de sensores, da rede e energéticas.

Com a realização deste trabalho pretende-se desenvolver um ambiente em que seja possível analisar diferentes cenários de simulação e vários tipos de aplicações. Com os resultados obtido, além de achar a melhor solução para o caso prático, pretende-se fazer uma análise do desempenho no encaminhamento de tráfego pelo protocolo.

As principais etapas a cumprir na realização desta simulação são as seguintes:

1. Criação de uma rede inicial de testes constituída por apenas dois nós *sender* e um nó *sink*;
2. A análise desta rede de testes permite, através do *collect view*, proceder a algum ajuste no código, se for necessário;
3. Carregamento da planta do edifício, previamente desenhado;
4. Análise da rede através do *collect view*;
5. Forçar alterações na rede, como taxa de sucesso da transmissão e criação de nó *sender* "incendiário";
6. Nova análise;
7. Obter conclusões devidamente fundamentadas através de dados e gráficos.

3.3 Sumário

Durante muitos anos, as entidades que prestam socorro em situações de emergência, nomeadamente incêndios, estavam dependentes de plantas de edifícios para a criar a abordagem mais correta face ao incêndio. Pode existir claro a experiência prévia de um simulacro onde é testada a operacionalidade do plano de emergência. O facto de poder haver informações incompletas pode causar algum atraso da chegada dos bombeiros ao local, existe ainda a questão mais remota de falsos alarmes.

As capacidades de monitorização e diagnóstico remoto de um sistema IoT ajudam os bombeiros a saber onde posicionar a equipa de resgate e preparar todos os seus recursos com antecedência. Um sistema IoT indica à equipa de resgate a localização de um detector de fumo e o último valor lido. E graças a estas novas tecnologias de baixo custo, tais como este tipo de dispositivos e a sua facilidade de implementação, a adoção em vários edifício está a tornar-se cada vez mais frequente.

Por fim, a grande quantidade de dados gerados por esta onda de dispositivos, podem ser tratados e analisados em uma solução *cloud*. Isto permite que dispositivos remotos, como *smartphones* tenham também acesso a essa informação a partir de qualquer lado.

Capítulo 4

Simulação de um caso prático

O edifício é inspirado no hotel *The Plaza*, construído em 1905. Uma simulação desta natureza também é aplicável a um edifíco que tenha ficado parado no tempo em termos tecnológicos.

A escolha do caso não é aleatório, trata-se de com base num determinado modelo, demonstrar os benefícios e potencialidades do IoT. Independentemente da sua infraestrutura é possível rapidamente implementar uma solução que visa melhorar a segurança, privilegiada pela interconectividade e praticidade.

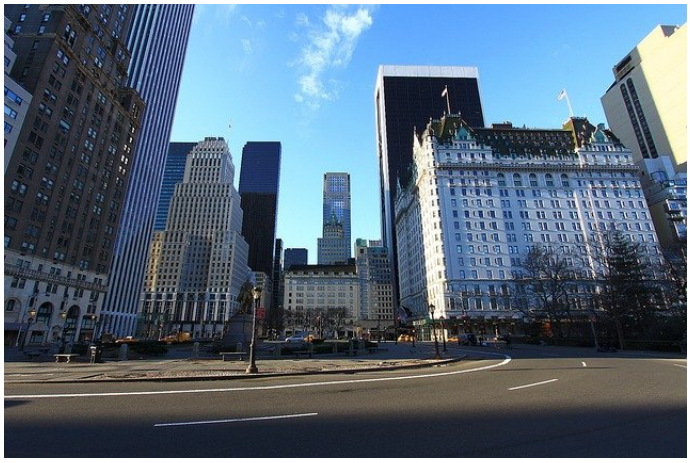

Figura 4.1: Hotel The Plaza, Nova Iorque

4.1 Detalhes da simulação

A simulação em si, será feita com recurso à ferramenta *collect-view* presente no simulador *cooja*, pois a partir daí é possível obter alguns gráficos interessantes de modo a validar esta simulação.

No ambiente de simulação existem apenas dois tipos de elementos. A rede é maioritariamente ocupada por emissores UDP, configurados no ficheiro `udp-sender.c`, estes dispositivos (*Tmote Sky*) têm a função de recolher a temperatura das salas em que estão instalados e comunicar essa informação pela rede. Depois existe o *Sink* UDP que funciona como *gateway* e pode ser configurado através do ficheiro `udp-sink.c`.

O projeto criado, com recurso aos ficheiros padrão do simulador permitem criar uma rede de raiz onde é feita a troca de informação entre cliente e servidor UDP, além disso, a configuração padrão facilita a comunicação RPL, onde são naturalmente tidos em conta os caminhos DAG e DODAG. A diretoria dos ficheiros mencionados assim como outros essenciais ao seu funcionamento é `contiki/examples/ipv6/rpl-collect`.

4.2 Escolha RPL

A escolha do protocolo RPL foi de certa parte justificada previamente em 3.1.1, onde é abordado na parte referente à comunicação e discutido no fim deste mesmo capítulo.

Como foi referido, a escolha deste protocolo deve-se essencialmente ao facto de ser indicado para este tipo de rede e cenário, o facto de vir incluído no sistema no qual é feito a simulação também foi um factor determinante, além das suas vantagens em relação a outros protocolos.

Não obstante, é pertinente abordar alguns pormenores deste protocolo assim como uma breve explicação do seu funcionamento.

Ao recapitular [6], pode-se constatar que as características de RPL fazem deste protocolo um pouco mais eficiente quanto à gestão de memória, o facto de ser *distance vector* dispensa a construção e manutenção de uma base de dados onde consta cada nó vizinho e rota. O RPL calcula um gráfico acíclico orientado para o destino (DODAG) com base numa função e num conjunto de métricas e restrições. No contexto do encaminhamento, um gráfico acíclico dirigido (DAG) é formado por uma série de nós e ligações, e cada ligação liga um nó a outro. O DODAG é uma topologia lógica construída sobre a rede física, o principal objectivo é encaminhar o tráfego de dados que é por vezes sujeito a determinados requisitos.

Estes requisitos consistem no cumprimento de alguns critérios a nível de métrica assim como algumas restrições. Objetivamente, pretende minimizar a latência na comunicação, por sua vez o aumento do sucesso quanto à entrega de mensagens é um objetivo a cumprir. As métricas são valores escalares, pode-se dizer que são parâmetros de entrada que permitem ao algoritmo a selecção do melhor caminho. As restrições referem-se a condições da rede, nomeadamente à possibilidade de excluir nós ou ligações específicas, tais como a exclusão de nós alimentados por bateria ou evitar ligações não encriptadas.

Numa rede RPL, um determinado nó pode ser membro de diferentes topologias lógicas, cada uma com um objectivo diferente, conhecido como "instâncias"em RPL.

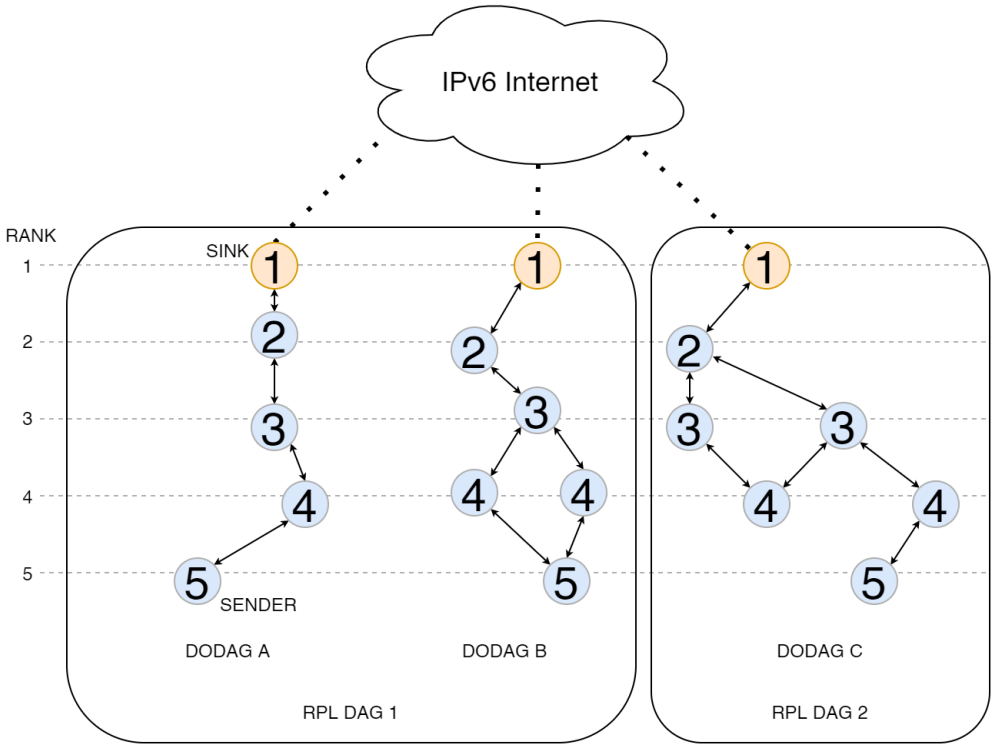

Figura 4.2: Exemplo de uma rede RPL composta por duas instâncias: uma com dois DODAGs e outra com apenas um

Uma instância(DAG) é um conjunto de DODAGs, ou um conjunto de vários nós que partilham um raio de ligação. Na figura 4.2 é possível observar a instância RPL DAG 1 constituída por dois conjuntos DODAG, A e B, à esquerda e à direita respetivamente. Um nó raiz DODAG é considerado um *router*, neste caso, denominado de *sink*, (elemento alaranjado). É classificado como mais importante, devido ao facto de ser um elemento central, daí, é-lhe sempre atribuída uma classificação de 1, sendo o 1, o valor de maior importância considerando uma espécie de *ranking*, o protocolo classifica os nós sensores restantes, *senders*, representados por um tom azulado, a partir do nó *sink*, quanto mais afasto do nó *sink* mais elevado será o seu identificador. No fundo, esta classificação indica a posição ou coordenadas do nó na hierarquia gráfica. Na figura 4.2 os nós tem números respectivos ao seu *ranking*, ao contrário do simulador em que o número representa o seu identificador.

O RPL tem características que o tornam adequado para redes instáveis e sujeitas a perdas [33]. Em primeiro lugar, é um protocolo proativo, ou seja, tem a capacidade de calcular caminhos alternativos em caso de algum inconveniente, enquanto protocolos reactivos dependem da troca de mensagens após a ocorrência de uma falha, só assim consegue determinar esse caminhos. Segundo, o RPL privilegia a reparação local em relação a uma revisão global da rede. As falhas são tratadas localmente, onde é escolhido um caminho alternativo, o que torna o protocolo o mais indicado para redes com ligações sujeitas a perdas.

4.3 Nó Sensor

Uma rede de sensores sem fios possui características bem específicas explícitas em 2.1.2, mas basicamente é um conjunto de vários sensores nós ou motes como também são conhecidos, que comunicam sem fios. Abordados em 2.1, estes dispositivos são o elemento tecnológico principal desta dissertação, tipicamente são de baixa potência, possuem capacidade de armazenamento e processamento, além de serem concebidos para comunicarem entre si sem fios a relativamente curtas distâncias.

Nas subsecções 4.3.1 e 4.3.2 são abordados nós sensores do tipo *sender* e do tipo *sink*.

4.3.1 *Sender*

A recolha de dados é realizada através de motes configurados como *sender*, com a finalidade de medir temperaturas em cada divisão do edifício. A cada nó sensor foi acrescentado algum código, de modo a que fosse possível a criação desses valores para efeitos de simulação. Dispositivos deste tipo possuem funções específicas de leitura e comunicação .

```
#include "sys/etimer.h"
static struct etimer et;
/*---------------------------------------------------------------*/
// todos os processos tem dois argumentos: nome e strname
PROCESS(send_sensor_info_process, "Print the Sensors Information");
// início automático dos processos assim como os seus argumentos
AUTOSTART_PROCESSES(..., &send_sensor_info_process, ...);
/*---------------------------------------------------------------*/
static int
get_temp(void) { //funcao de temperatura
return (24); //retorna 24 graus celsius de padrao
}
/*---------------------------------------------------------------*/
//processo: pode ser chamado a qualquer momento e estar dependente de eventos
PROCESS_THREAD(send_sensor_info_process, ev, data)
{
  PROCESS_BEGIN(); //início de processo
  etimer_set(&et, CLOCK_SECOND*10); //timer configurado para 10 segundos
  while(1) {
        PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&et));
        if(etimer_expired(&et)==1) {
        printf("Temperature: %d \n", get_temp());
        etimer_restart(&et); //reiniciar timer
        }
   }
 PROCESS_END(); //fim de processo
}
```

Listagem 4.1: Código adicionado em udp-sender.c

Na presente demonstração, apesar de inicialmente ter sido implementada a leitura de valores através do módulo sht11, esta funcionalidade não foi usada devido a algumas incompatibilidades com a protocolo de comunicação, assim como a função *etimer*.

Há que ter atenção à gestão energética destes dispositivos, para tal os valores só serão transmitidos esporadicamente. Assim, inicialmente tem de se declarar e implementar o processo principal, que funcionará assim como uma espécie de aceno assim que se der início à simulação. Visto não estar integrado o módulo `sht11_init()`, os valores são gerados de um modo estático a partir da função `get_temp()`. Por curiosidade, sht corresponde a *Sensor Humidity Temperature*, e 11 corresponde a onze milisegundos que é geralmente o tempo de reacção após a sua chamada.

Caso esse módulo em questão fosse utilizado, seriam necessários alguns cálculos matemáticos elementares e necessários à interpretação correta dos valores lidos, seriam facilmente aplicados em `C` de acordo com as especificações do *datasheet* referente ao *Tmote Sky*.

4.3.2 *Sink*

As redes de sensores sem fios podem ter vários tipos de topologia, apesar de a do caso prático poder estar sujeita a alterações, uma vez que os dados percorrem vários *hops* até chegar ao nó central (*sink*), é considerada *mesh*. O protocolo de encaminhamento tem a função de encontrar o melhor caminho para transportar a informação de nó origem (*sender*) até ao nó central, a partir daí pode tratar a informação de diversas maneiras.

O código permaneceu intacto relativamente à sua configuração padrão.

4.3.3 Rede de inicial de teste

Durante a programação dos sensores foi importante ir testando exaustivamente as alterações realizadas, especialmente em ambientes de desenvolvimento a cada alteração implicava compilar e posteriormente orquestrar uma pequena rede de sensores. O ambiente de trabalho neste caso era a diretoria `contiki/examples/ipv6/rpl-collect/`, aqui foi alterado essencialmente o ficheiro `udp-sender.c`.

Uma vez a compilação tenha sucesso, esta fase de testes consistia em dispor dois sensores, 1 e 2, e um *sink* com o identificador 3, posto isto pretende-se analisar o comportamento da rede. Para esta fase de testes os requisitos de comportamento são o envio consistente de temperaturas por parte dos sensores e a consequente resposta do *sink*. A obtenção da resposta do *sink* é fulcral a este projeto, só assim se consegue obter os gráficos de análise de toda a envolvente da rede.

Na figura 4.3 é possível observar a disposição da rede de testes em cima do lado esquerdo. Aqui, existem algumas opções de visualização, realçando *Mote IDs*, *Type* e *attributes*, e *Radio traffic* e *environment*. Em cima do lado direto pode-se obervar os botões de controlo da simulação, finalmente em baixo o *output* dos dispositivos em toda a rede.

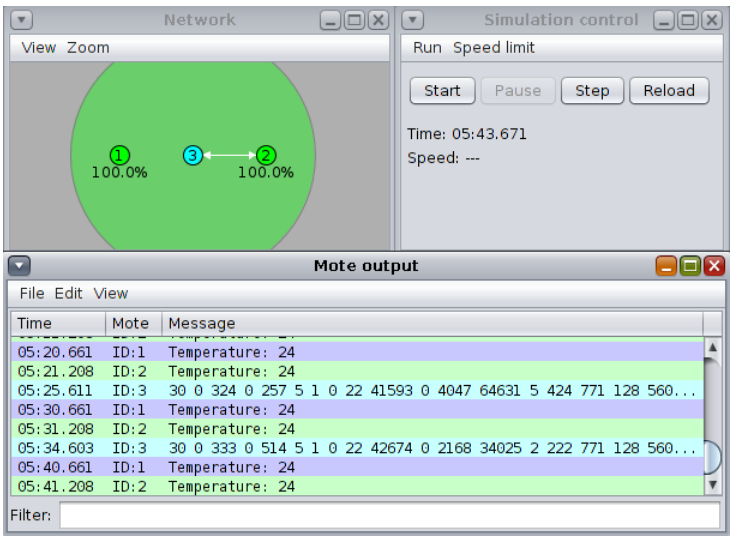

Figura 4.3: Rede inicial de testes

4.4 1º Cenário

A implementação de uma rede de sensores neste hotel exige uma sequência de passos [34], de modo a dispor os sensores pela planta do edifício. Na tabela 4.1, é possível conferir os detalhes da simulação, o que proporciona uma melhor percepção desta caso prático. Desde logo será possível comprovar a facilidade e interconectividade desta simulação, caso fosse implementada na vida real.

Após abrir a linha de comandos da máquina virtual, entra-se na diretoria do simulador, `contiki/tools/cooja`. Aqui é executado o comando `ant run`. Em alguns segundos, surge o ambiente de trabalho do simulador *cooja*. De seguida:

- Entrar no menu: *File* > *New Simulation*;
- Dar um nome à simulação e clicar *Create*.

No entanto, a simulação criada não tem nenhum nó. Para adicionar um nó, começa-se por adaptar ao caso prático, em 4.1. Só depois de carregados e compilados pode-se adicionar à simulação. No *cooja* existem vários tipos de nós. Cada tipo pode ter várias configurações e evocar diferentes funções.

De modo a adicionar os nós ao ambiente de simulação:

1. Na barra do topo, procurar *Motes*>*Add motes*>*Create new mote type*>*Sky mote*...
2. Em *Browse*, procurar nas diretorias o nó previamente programado;
3. Uma vez seleccionado, compilar e criar;
4. Seleccionar a quantidade e a disposição.

Uma vez posta em prática estes procedimentos, existem condições para dispor os vários sensores de forma a construir a rede. Na figura 4.5 está o primeiro esboço, que de um modo geral ajuda a ter uma ideia da primeira disposição dos sensores e, naturalmente, o número de sensores e *sinks* neste caso prático. Mas antes é importante ter noção do seu comportamento e como ocorre a troca de informações numa rede deste género.

Ao observar a figura 4.4 é possível ter uma ideia que a informação é reencaminhada pelos sensores vizinhos até chegar ao nó *sink*. Cada nó tem um raio de transmissão, cada nó vizinho que estiver dentro desse raio pode receber e reencaminhar esses dados. O nó *sink* acusa a recepção da informação através de uma mensagem.

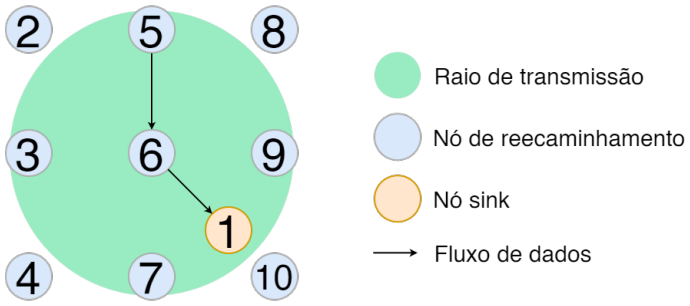

Figura 4.4: Topologia de simulação

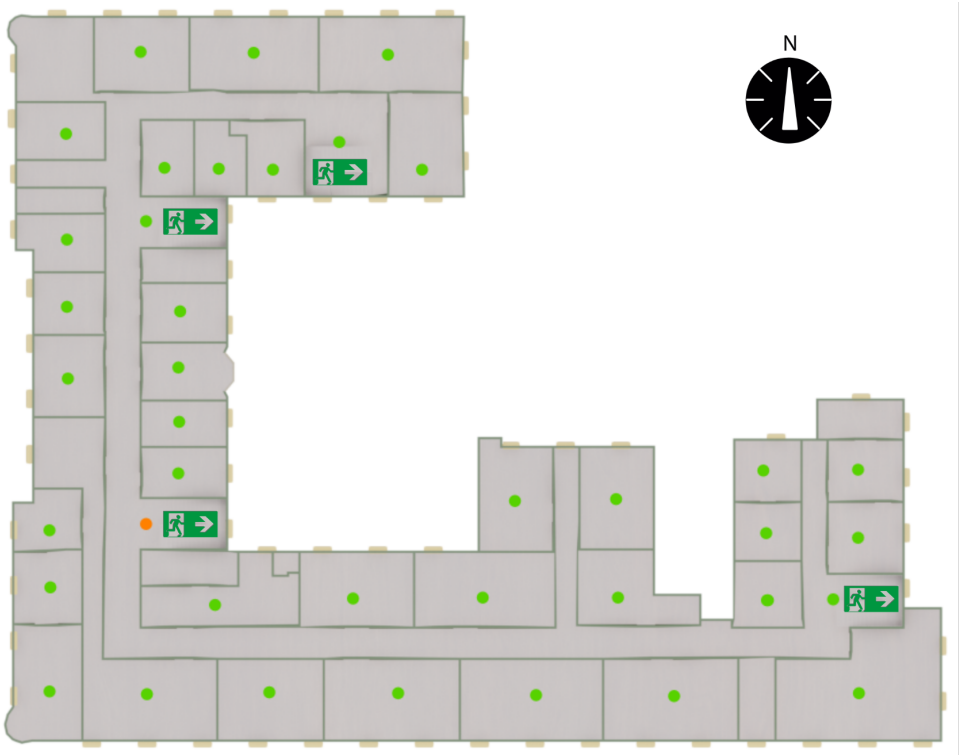

Figura 4.5: Planta do hotel com disposição de sensores

Tabela 4.1: Configuração de Contiki e Cooja

Parâmetros	Características
Modelo de canais Wireless	UDGM Model with Distance Loss
Raio de transmissão	10 m
Separação entre nós	Variável
Área	60x60 m^2
Densidade	Aprox. 2 sensores por $10m^2$
Número de sensores	38 sensores e 1 sink
Tipo de sensores	Tmote Sky
Camada de rede	μIPv6 +6LoWPAN
Camada MAC	CSMA + ContikiMAC
Interface Rádio	CC2420 2.4 GHz IEEE 802.15.4
Rácio de sucesso tx/rx	1.0 / 1.0
Tempo de simulação	8h

4.4.1 Configuração da Rede

De modo a um maior enquadramento e percepção foi feita uma tabela, onde constam alguns detalhes da simulação. A tabela 4.1, pretende de um modo simples e direto definir os parâmetros iniciais.

Entretanto, na figura 4.6 é visível a disposição dos sensores pela rede. As posições destes sensores já estão de acordo com as dimensões do edifício em estudo.

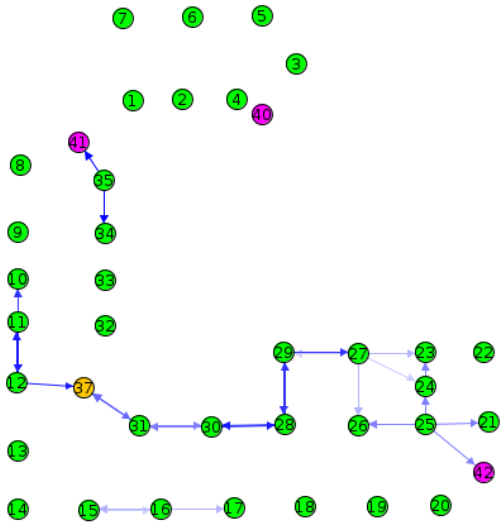

Figura 4.6: Primeiro teste

Além de alguma atividade na rede é possível distinguir os tipos de sensores, os identificados em rosa encontram-se perto das saídas de emergência. Os sensores verdes têm exatamente a mesma configuração dos rosa, ou seja, são do mesmo tipo, a cor diferente deve-se apenas a questão da identificação das saídas de emergência. Existe depois o nó *sink* (alaranjado), para aqui, é encaminhada toda a informação da rede, neste caso, valores de temperatura.

4.4.2 Resultados de Simulação

Após o primeiro teste com a duração de 8 horas é possível verificar as ligações criadas entre nós vizinhos durante esse período. A figura 4.7 ilustra além destas ligações criadas o fluxo e o sentido da informação enviada. O destino final das temperaturas é o nó *sink*, colocado estrategicamente no edifício.

Inicialmente, durante os primeiros testes, existiam 4 nós *sinks*, no entanto esta organização forçava a topologia em *cluster*. Especificamente neste caso, não existe interesse nessa topologia uma vez que não permitia estudar a rede no seu todo, a ferramenta *collect view* apenas permite recolher dados de um nó *sink* de cada vez, o que não impede de filtrar os nós que se deseja nessa análise.

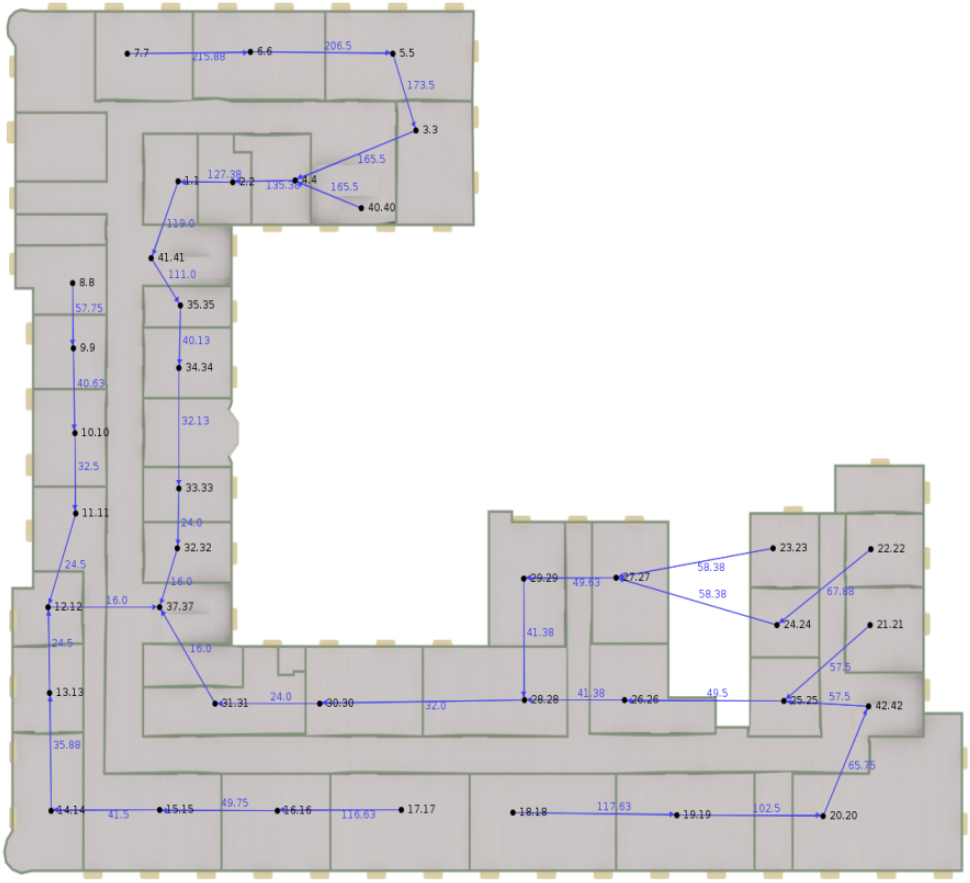

Figura 4.7: Ligações criadas entre nós

A ferramenta *collect view* permite ter uma visão heurística da rede de sensores. Uma vez acedido o *collect view* através do nó *sink*, é possível obter uma série de gráficos fruto dos dados recolhidos ao longo da simulação. Primeiramente pode-se observar na 4.8 o número de saltos.

Muitas vezes referido como *hop count*, é a forma mais tradicional de métrica, qualquer protocolo de encaminhamento em redes de sensores sem fios lida de algum modo com o número de saltos. Geralmente permite achar o caminho mais curto entre origem e destino através do número mais baixo de *hops*.

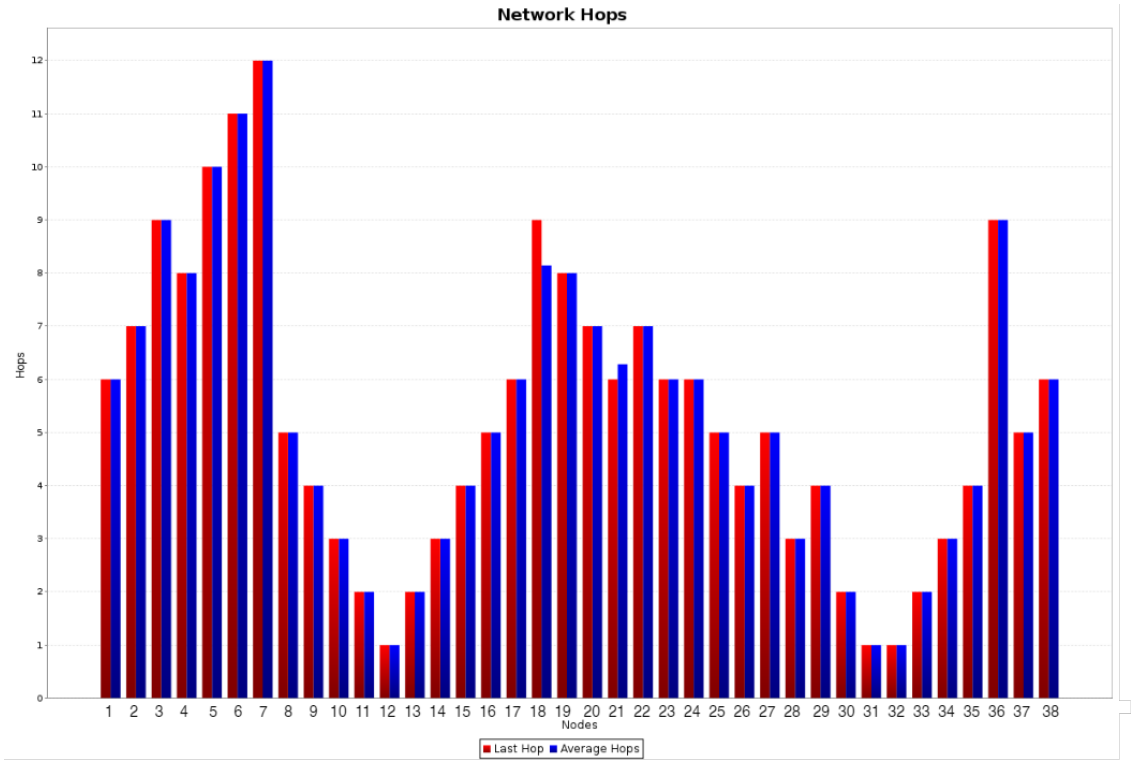

Figura 4.8: Network hops por nó

Ao consultar este gráfico (4.8), e tendo o conhecimento prévio da disposição da rede (4.7), pode-se concluir imediatamente que o sensor com o identificador 7 é o mais longínquo do nó *sink*, onde o valor da temperatura dá 12 saltos até chegar ao seu destino, e que, os sensores 12, 31 e 32 são vizinhos do *sink*, (quanto menores as barras, mais perto os sensores se encontram do nó *sink*), até agora informações que dispensavam qualquer tipo de gráfico.

No entanto, além deste gráfico proporcionar uma visão diferente do que já se sabia, é possível verificar que que a informação proveniente dos nós 18 e 21 nem sempre fizeram o mesmo percurso pois o *last hop* não corresponde a *average hops*, uma característica dos nós móveis.

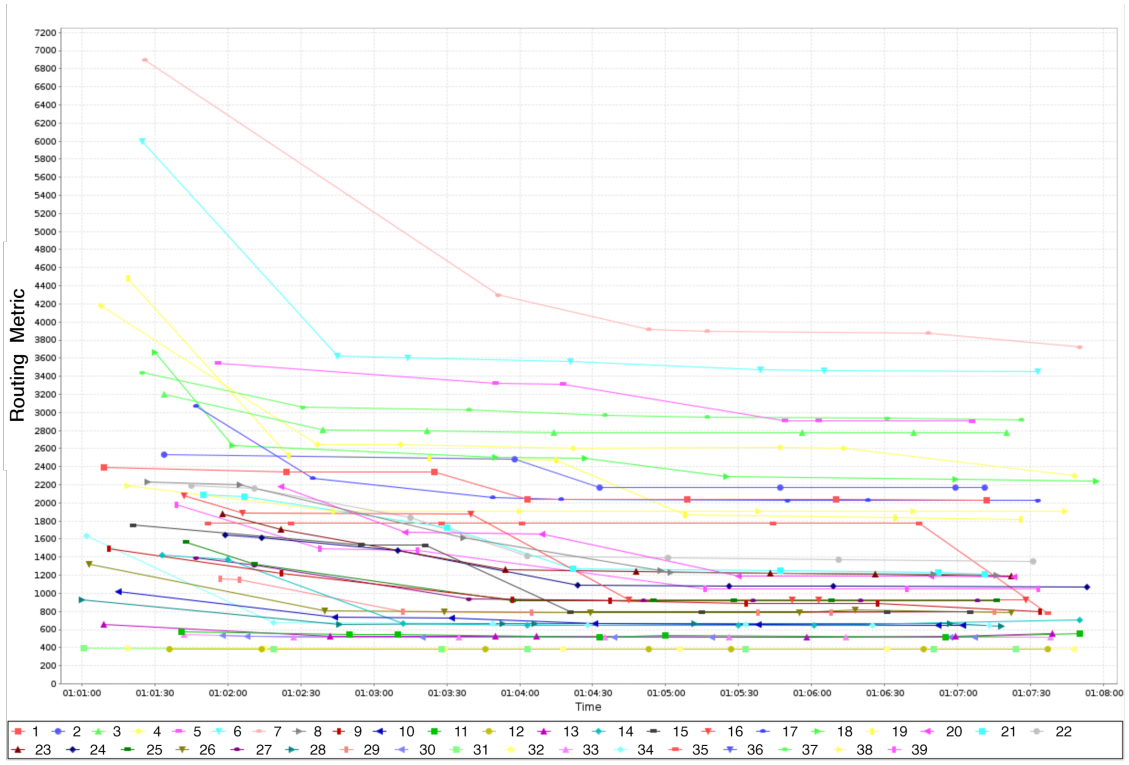

Figura 4.9: Métrica de encaminhamento de todos os sensores

A métrica de encaminhamento é fruto do número de saltos, é o que permite ao longo da simulação a rede determinar os melhores caminhos para todos os nós. É usada para determinar se uma rota deve ser escolhida em detrimento de outra, existe uma tabela de encaminhamento que armazena todas as rotas possíveis.

Particularmente nas redes de sensores sem fios, a métrica é facilmente implementada através da contagem de saltos, o protocolo de encaminhamento limita-se a incrementar o número de saltos e posteriormente escolher o caminho que tem menos saltos.

Na figura 4.9 são visíveis os três nós vizinhos do nó *sink* (12, 31 e 32), possuem uma métrica baixa e constante, enquanto os nós mais afastados (7,6 e 5) são iniciados com uma métrica mais elevada e requerem alguma aprendizagem da rede de modo a diminuírem. Apesar de parecer algo confuso, o essencial a reter deste gráfico é a diferença entre os nós mencionados acima, entre estes, é possível observar todos os outros maioritariamente compreendidos entre 500 e os 3000 na escala de *routing metric*, alguns inicializam com valores bastante elevados, mas depressa baixam e estabilizam.

Daqui advém ETX (*Expected Transmission Count*), não é mais que a métrica de encaminhamento que permite encontrar caminhos de alto rendimento neste tipo de redes. Não é representado o gráfico devido a muitas semelhanças com a figura 4.9 e em nada acrescentar de relevante.

O *Beacon Interval* ou intervalo de sinalização é definido pelo tempo entre duas *frames* consecutivas durante a comunicação de dois sensores, é composto por uma parte ativa, também conhecida como *superframe*. Este desenho bastante sugestivo, adaptado de [35], permite entender rapidamente esta ocorrência.

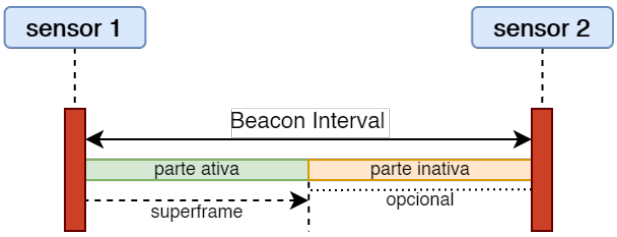

Figura 4.10: Beacon Interval

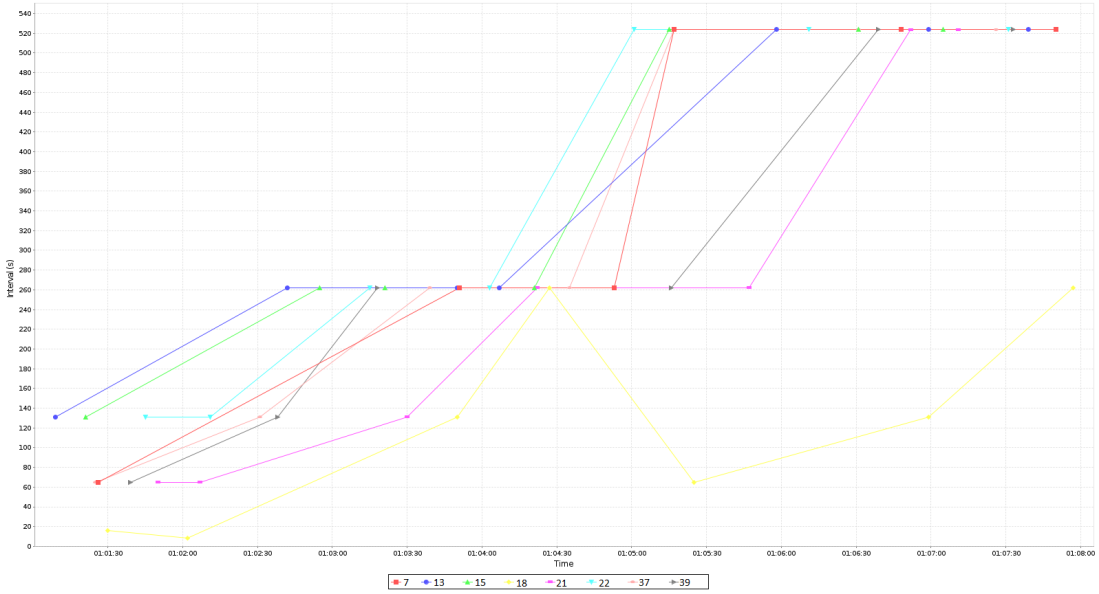

Figura 4.11: Beacon Interval de alguns sensores

O gráfico em 4.11 mostra o intervalo de sinalização. Ao contrário dos gráficos anteriores, neste estão apenas representados alguns sensores, pois este ficaria algo confuso.

Logo à partida é de notar que o nó 18, representado a amarelo mostra bastante inconsistência, essencialmente porque se encontra longe do nó *sink*, e possui várias rotas com pesos iguais ou semelhantes. Pode ainda ser um fator alheio ao simulador, como por exemplo na demora da ligação rádio.

Os restantes nós também possuem algumas particularidades, nomeadamente os nós 21 e 22 terem várias rotas possíveis, e os nós 37, 39 atuarem como nós de transição, inicialmente no mesmo local de outros *sinks*. O nó 7 devido ao facto de ser o que se encontra mais longe do *sink* tem intervalos maiores.

A nível energético

Por fim, outro fator sujeito a análise e em que poderá ser possível ver alterações na rede de sensores é o estudo da componente energética. Seguidamente serão apresentados dois gráficos e respetivamente feita a sua análise de acordo com a configuração da rede.

Na figura 4.12 mostra o consumo médio energético de todos os sensores. Antes de passar à análise, é importante identificar as fontes de consumo energético de um sensor, representados por cores. A vermelho está o consumo respetivo ao *Low Power Mode* (LPM), valor padrão para todos os sensores, a azul o consumo realizado pelo processamento do sensor. A barra de consumo médio fica completa ao acrescentar os consumos rádio, tanto para receção como transmissão, representados por verde e amarelo, respetivamente.

Posto isto, ao observar o gráfico, é possível ver imediatamente que o sensor 30 é o que consome mais energia, juntamente com os nós 28 e 31, estes sensores estão sujeitos a uma grande quantidade de tráfego proveniente da ala Este do edifício, essa quantidade de tráfego implica mais capacidade de processamento, e naturalmente, uma maior taxa de receção/transmissão desse mesmo tráfego.

Por fim, é de notar que nós como 7 e 36 limitam-se a enviar os seus valores de temperatura, por estes passa pouca ou nenhuma informação proveniente de outros sensores.

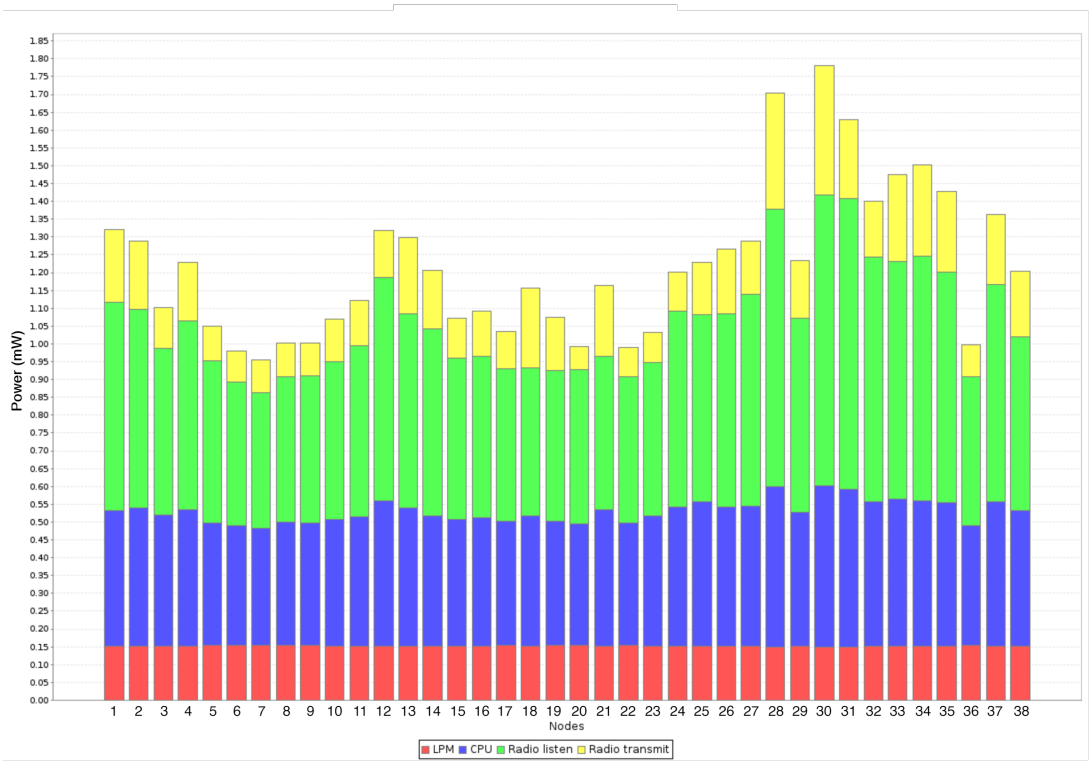

Figura 4.12: Consumo médio energético por sensor

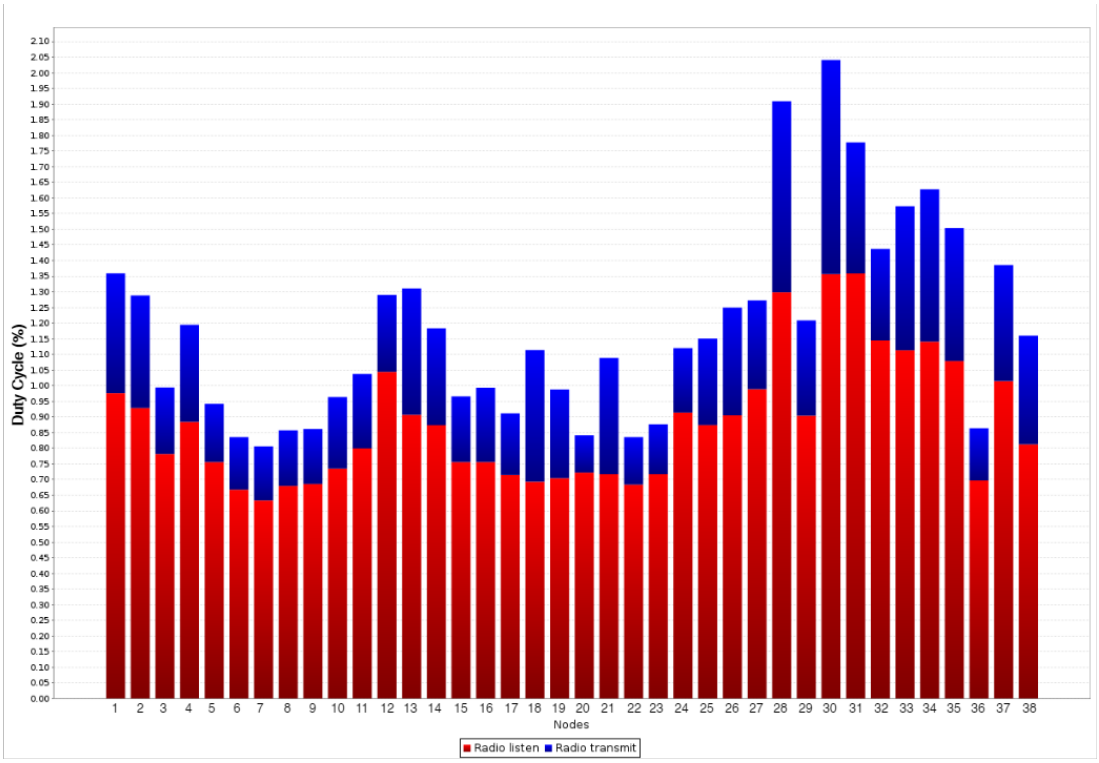

Figura 4.13: Duty Cycle

O Duty Cycle trata-se da fracção de tempo em que um determinado equipamento se encontra ativo. Tecnicamente é o rácio do uso da componente rádio, ou seja, os tempos de uso tanto de escuta como de transmissão em relação ao tempo em que um determinado sensor se encontra ativo, usualmente o nó *sink* não é considerado visto que é suposto estar sempre ativo [36]. O Duty Cycle é calculado (4.1) para um nó somando os tempos de transmissão e receção rádio, e dividindo pelo total de tempo ativo, além do tempo de processamento é considerado para este cálculo o modo LPM.

Através do gráfico 4.13, é possível distinguir sensores que se limitam a enviar as suas temperaturas, dos que para além disso, estão localizados em sítios onde são obrigados a encaminhar os dados de outros sensores, o que implica a um maior uso do módulo rádio.

$$DutyCycle = \frac{Transmissao + Rececao}{Processamento + LPM} \tag{4.1}$$

4.4.3 Validação

A análise é validada através do *output* dos sensores. Umas vez feita a programação dos sensores e posteriormente a configuração da rede, a melhor forma de verificar o funcionamento correto da rede é através da janela *mote output*, isto é, a leitura de saída dos sensores. Assim que iniciada a simulação começam a surgir mensagens provenientes dos sensores, e quando se verifica o correto envio de temperaturas com os *timers* estipulados e as respostas do nó *sink* é sinal que tudo está a correr bem, na figura 4.14 pode-se observar o *mote output* deste primeiro cenário.

Time	Mote	Message
01:10.865	ID:10	Temperature: 24
01:10.903	ID:12	Temperature: 24
01:10.945	ID:29	Temperature: 24
01:10.972	ID:22	Temperature: 24
01:10.999	ID:26	Temperature: 24
01:11.017	ID:9	Temperature: 24
01:11.024	ID:5	Temperature: 24
01:11.028	ID:19	Temperature: 24
01:11.068	ID:37	30 0 69 0 3341 1 2 0 22 8922 0 5818 65498 701 789 3084 214 657 1 131 252 1 133 182 65535 6553...
01:11.096	ID:41	Temperature: 24
01:11.118	ID:17	Temperature: 24
01:11.192	ID:13	Temperature: 24
01:11.208	ID:3	Temperature: 24
01:11.215	ID:21	Temperature: 24
01:11.286	ID:32	Temperature: 24
01:11.308	ID:23	Temperature: 24
01:12.424	ID:37	30 0 71 0 2313 1 4 0 22 9080 0 2682 33607 305 320 2570 556 1493 1 65 252 1 133 182 65535 6553...
01:16.645	ID:37	30 0 75 0 2570 1 3 0 22 9664 0 2997 35626 340 371 2827 376 1016 1 131 252 1 133 182 65535 655...

Figura 4.14: Output de mensagens da rede

Como ferramenta de comparação foi também feita uma extração dos pacotes do Contiki e analisada no *Wireshark*, visível no gráfico em 4.15, aqui é possível observar uma maior atividade na fase inicial, onde se está a criar a rede propriamente, isto é, os sensores conhecerem os seus nós vizinhos. Após algum tempo pode-se observar uma maior estabilidade, onde apenas ocorre o envio de temperaturas e respostas do nó *sink*.

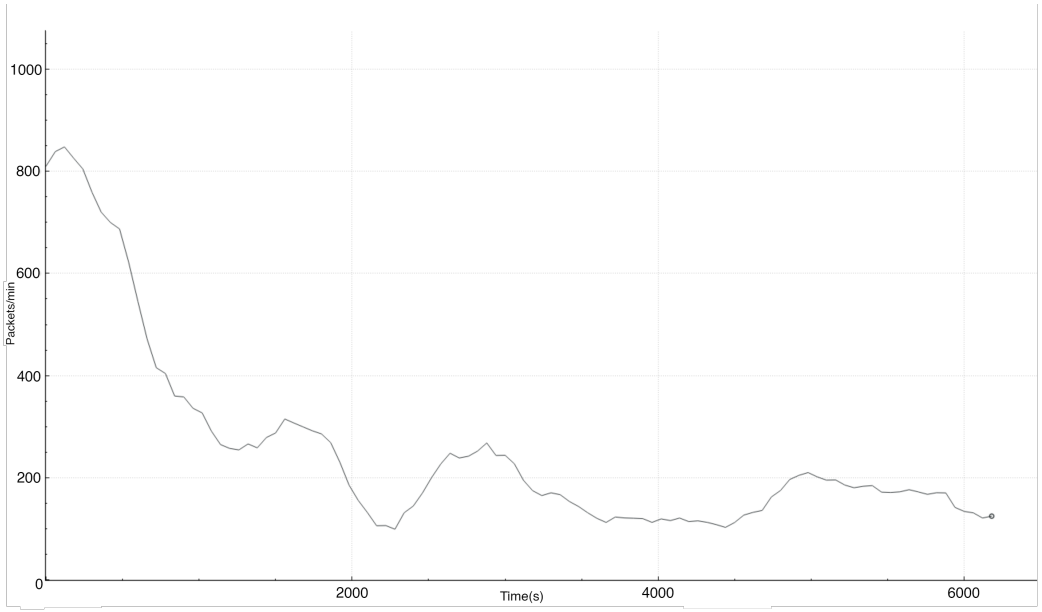

Figura 4.15: Atividade da rede em pacotes por minuto

4.5 2º Cenário - Incêndio

Para o segundo cenário, foram programados alguns sensores de modo a simular um incêndio. No código apresentado em 4.2 apenas foram alterados o valor da temperatura e a frequência das suas leituras.

```
static int
get_temp(void) { //funcao de temperatura
return (59); //retorna 59 graus celsius
}
/*---------------------------------------------------------------*/
PROCESS_THREAD(send_sensor_info_process, ev, data)
{
  PROCESS_BEGIN(); //início de processo
  etimer_set(&et, CLOCK_SECOND/4); //timer configurado para 1/4 de segundo
  while(1) {
        PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&et));
        if(etimer_expired(&et)==1) {
        printf("Temperature: %d \n", get_temp());
        etimer_restart(&et); //reiniciar timer
        }
   }
 PROCESS_END(); //fim de processo
}
```

Listagem 4.2: Código alterado em udp-sender-fire.c

Enquanto na figura 4.16 é possível observar a localização dos sensores alterados, representados a amarelo.

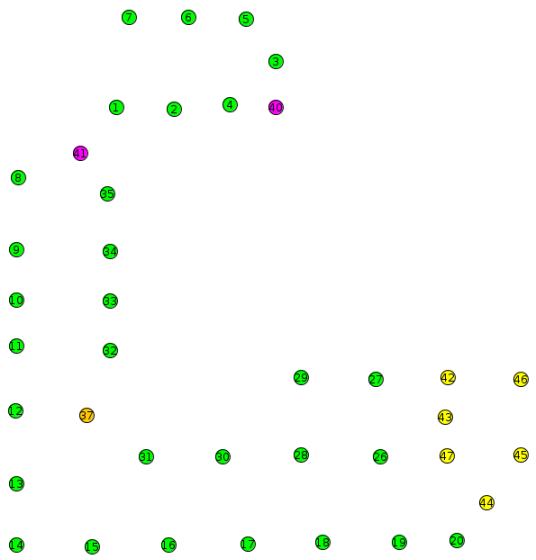

Figura 4.16: Configuração da rede - 2º Cenário

4.5.1 Análise

Após um estudo profundo e inúmeras simulações, as alterações mais significativas na rede foram observadas a nível energético. Devido a este facto, para análise deste cenário e para efeitos de comparação não serão considerados *network hops*, métricas nem *beacon interval*, uma vez que a disposição e configuração da rede é exatamente a mesma, assim como o número de sensores.

Neste caso não está definido qualquer tipo de alarme em caso de incêndio, no entanto é possível definir vários alarmes ou outro tipo de mensagens. Tudo isto é visível no *mote output* em tempo real durante a simulação. A única alteração é na programação de alguns sensores.

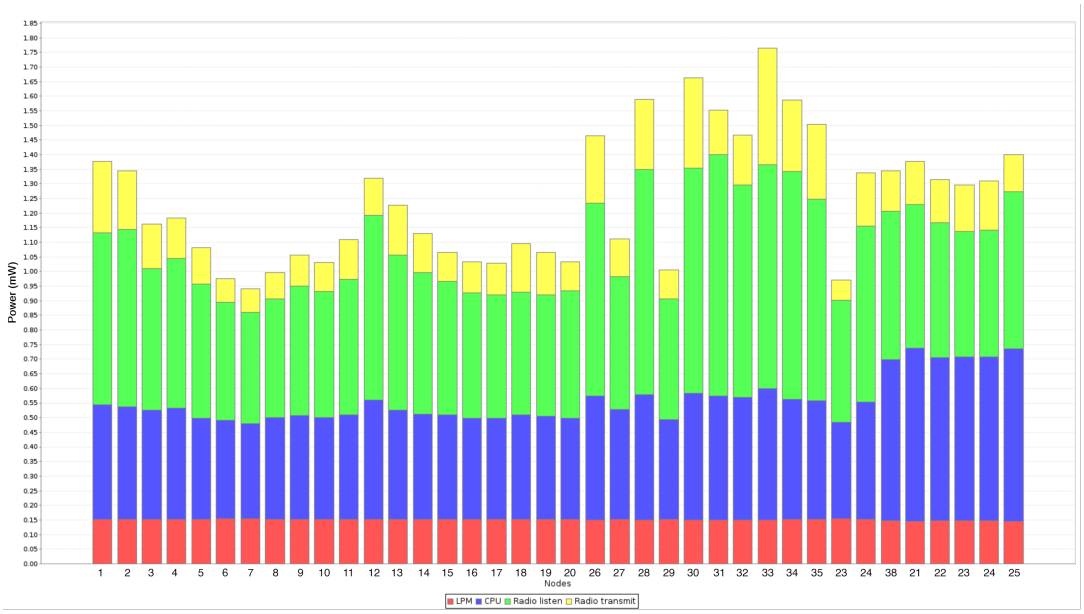

Figura 4.17: Consumo médio em caso de incêndio

Ao iniciar a análise deste cenário pela figura 4.17, pode-se novamente constatar, tal como no primeiro cenário, que os sensores mais próximos do nó *sink* são obrigados a reencaminhar os pacotes dos mais longínquos, logo, estão sujeitos a um maior consumo energético. No entanto, ao ignorar estes sensores em particular, existem outros que se destacam visualmente pelo seu elevado consumo médio energético, onde se incluem os sensores programados em 4.2 para criar este novo cenário de incêndio.

Ao analisar apenas as barras do gráfico desta gama de novos sensores, os seis sensores mais encostados à direita do gráfico (porque foram adicionados posteriormente à simulação), é discriminado o consumo dos seus CPUs (*Central Process Unit*), donde se conclui que este valor é mais elevado para estes novos sensores em relação aos restantes.

Adiante em 4.5.2 são comparados diretamente os dois cenários quanto aos seus níveis de consumos energéticos.

4.5.2 Comparação de simulações

As primeiras diferenças entre os dois cenários verificaram-se após a extração de dados referentes à comunicação rádio (*radiolog*), os dados recolhidos ao longo da simulação foram então abertos no *wireshark*. Este *software* permite analisar o tráfego de rede, tem uma funcionalidade que possibilita a filtragem de vários protocolos. O gráfico gerado é totalmente configurável. Ao contrário de da figura 4.15, o período SMA (*Simple Moving Average*) foi alterado de 100 ms para 50 ms, o que permite uma menor suavização na curva e consequentemente uma melhor e mais perceptível análise. Quanto à escala eixo de y manteve-se em pacotes por minuto, assim como o tempo de simulação em segundos, eixo de x.

Essencialmente na figura 4.18 são analisados os pacotes, a azul claro é tido em conta a média do volume dos pacotes UDP, enquanto a linha preta e a linha vermelha representam a troca de pacotes 6LoWPAN ao longo das simulações no cenário 1, em que tudo está em condições normais, e no segundo cenário em que decorre um incêndio respetivamente.

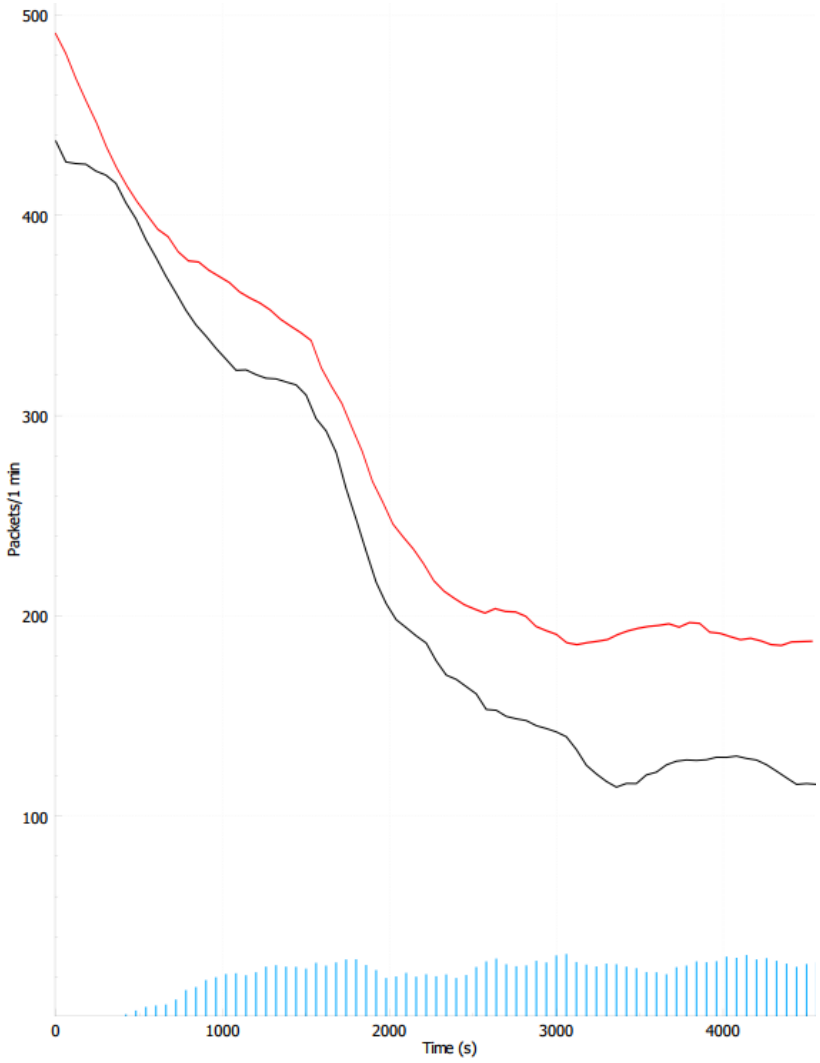

Figura 4.18: Comparação entre os dois cenários no wireshark

Uma vez extraídos os dados dos sensores nos dois cenários, encontram-se reunidas as condições para poder comparar os consumos energéticos entre os sensores referentes ao edifício em que tudo está normal, e quando os mesmos enviam temperaturas altas a uma frequência maior.

Tabela 4.2: Tabela de consumos energéticos dos dois cenários

	Cenário 1				Cenário 2			
Sensor ID	**CPU Pwr**	**Listen Pwr**	**Transmit Pwr**	**Power Total**	**CPU Pwr**	**Listen Pwr**	**Transmit Pwr**	**Power Total**
1	0,37	0,43	0,08	1,03	0,55	0,51	0,14	1,35
2	0,39	0,55	0,11	1,20	0,59	0,49	0,15	1,38
3	0,41	0,52	0,15	1,23	0,59	0,54	0,13	1,40
4	0,34	0,41	0,08	0,99	0,56	0,43	0,17	1,31
5	0,38	0,43	0,20	1,16	0,56	0,43	0,16	1,30
6	0,38	0,49	0,18	1,20	0,56	0,46	0,15	1,31

Da informação extraída do simulador *cooja*, em *node info*, foram filtrados os dados pretendidos dos dois cenários e apresentados na tabela 4.2.

Na figura 4.19, está representado graficamente apenas o consumo das unidades de processamento respetivamente nos dois cenários. É notório que o consumo aumenta substancialmente quando a rede está sujeita a uma maior atividade. De um modo genérico, o eixo de y apresenta várias localizações de sensores, uma vez que quando criada a segunda simulação, e apesar da mesma configuração de rede, os sensores alterados ficam com um identificador diferente. Deste modo, optou-se por esta designação, cada localização representa dois identificadores em cenários distintos, facilitando a comparação dos mesmos.

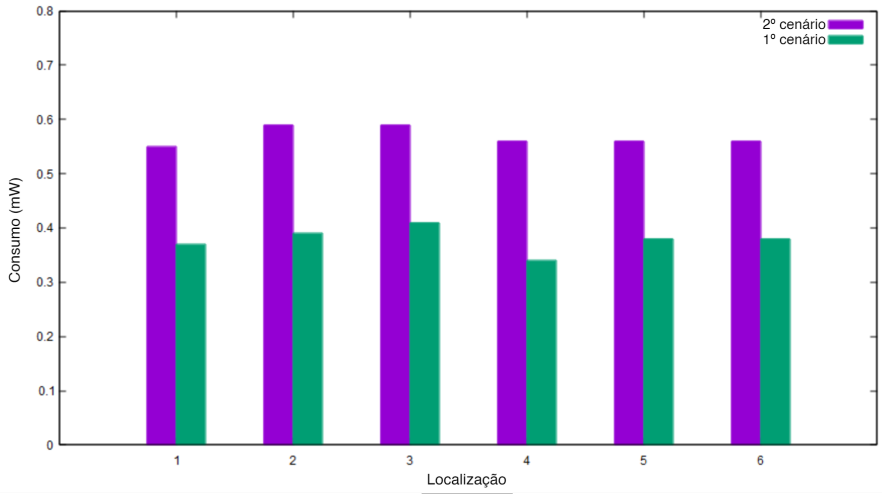

Figura 4.19: Comparação de consumo por processamento entre os dois cenários

Na tabela 4.2 não são apresentados os valores dos consumos respetivos ao LPM, uma vez além destes serem sempre iguais, não existe variação entre cenários.

Os consumos respetivos à receção e transmissão variam entre sensores e cenários, sendo mais elevados em alguns casos, não necessariamente devido a uma maior atividade na rede.

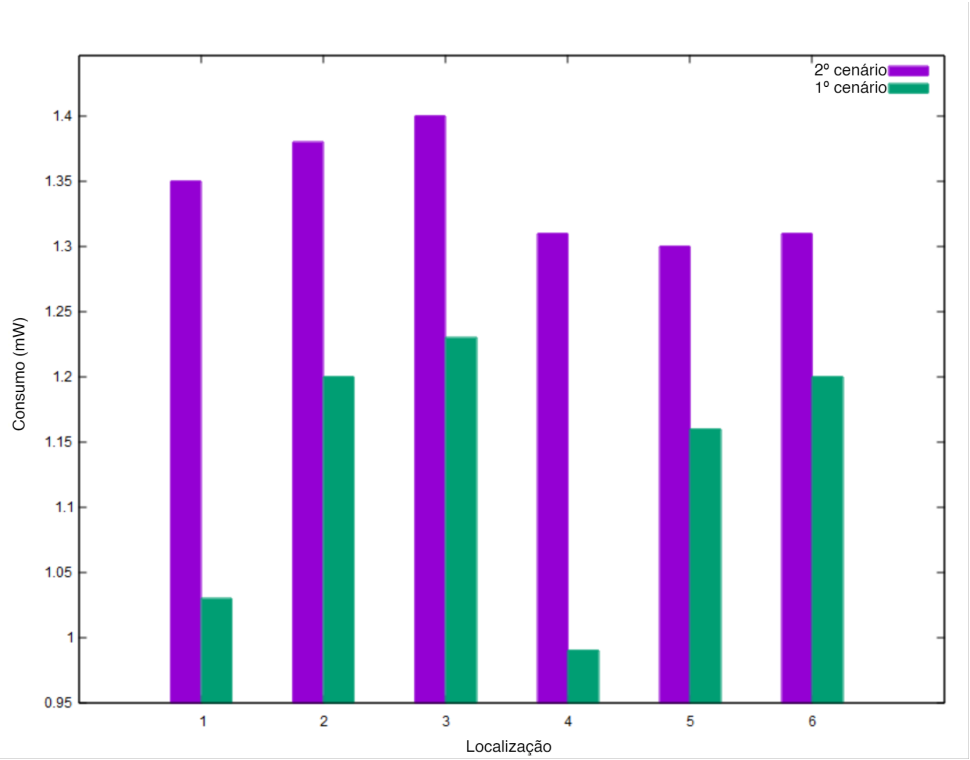

Figura 4.20: Comparação de consumo total entre os dois cenários

No final de contas, o consumo total do segundo cenário é consideravelmente mais elevado em relação ao primeiro.

O gráfico 4.20 ilustra a soma de todos os consumos, donde se conclui que naturalmente uma maior atividade na rede representa um maior consumo energético, relembrando que estes sensores estão sujeitos ao reencaminhamento de tráfego proveniente de outros sensores.

Capítulo 5

Conclusões e Trabalho futuro

5.1 Discussão de Resultados

Neste ambiente de simulação existiam inúmeras possibilidades, quanto à configuração da rede e à programação dos sensores. O simulador *cooja* é extremamente versátil podendo integrar com vários tipo de sensores reais e com vários protocolos de comunicação já pré-definidos, existindo a possibilidade de adaptação. Como resultado permite obter simulações extremamente fiáveis à realidade.

Com a configuração apresentada no capítulo 4 e através das simulações realizadas verifica-se alguns pontos interessantes, o comportamento dos sensores e do seu ambiente, a importância do protocolo de comunicação, podem ser exploradas algumas limitações da rede e do protocolo, nomeadamente a taxa de sucesso de entrega e receção de pacotes, raios de transmissão, em metros, entre outros. Daí, ter a sensibilidade de entender se as condições são as mais corretas para os que se pretende concluir.

Em relação aos sensores, das várias possibilidades existentes, o *Tmote Sky* como referido em 3.1.2, possui características que fazem deste o elemento responsável pela recolha de dados. Ao consultar o *datasheet* [37] referente a este sensor, verifica-se que tanto podem ser aplicados em ambientes fechados e uma distância até 50 metros, como em ambientes exteriores com distâncias de até 125 metros entre sensores. Concretamente neste caso prático, no simulador *cooja* o raio de transmissão foi reduzido até um máximo de 10 metros fazendo com que cada sensor tenha em média 2,53 sensores vizinhos. Por um lado, cada sensor tem um consumo energético mais reduzido, por outro lado implica um maior encaminhamento de tráfego, ao observar a figura 4.8 é possível constatar que o sensor com o identificador 7 percorre 12 *hops* até ao nó destino, aqui, os pacotes provenientes desse sensor, atravessam grande parte do edifício.

5.2 Perspectiva de futuro

Em termos de trabalho futuro, é necessário entender as características de cada sensor. No entanto, a partir do momento que tem um IP (*Internet Protocol*) associado, qualquer sensor pode integrar estrutura de gestão de risco de desastre dos dias de hoje. O IoT é o principal agente dos *smart buildings*, este eleva os sensores para a nova geração e proporciona um melhor envio e tratamento dos dados recolhidos, no caso desta dissertação estes dados são leituras de temperatura.

As empresas e entidades têm vindo a ter um cuidado especial com a segurança das pessoas através da transformação digital, observando o mundo físico de outra perspectiva, tirando proveito da visualização de dados.

A estrutura proposta em 3.4 visa integrar vária tecnologias, esta integração pode ser feita através da *cloud*, permitindo vários pontos de acesso e uma maior eficiência.

A disponibilidade dos dados para diversos fins é visto hoje como uma comodidade, torna várias tecnologias em apenas um ecossistema. Neste caso em concreto, existem pessoas que frequentam o edifício, as autoridades de resgate e o *smart building*, a partir daqui pode-se ter várias funcionalidades para o bem comum.

Além da finalidade principal, uma estrutura que facilita o resgate de pessoas em caso de emergência, existem outras, nomeadamente, controlo de consumo energético do edifício, gestão de ocupação de espaços, aqui é importante ter em atenção às leis de RGPD, além destas metas de um modo geral pretende-se cada vez mais melhorar a experiência de quem frequenta um *smart building*.

5.3 Conclusão

Foi uma jornada exigente mas bastante enriquecedora, todo o trabalho desenvolvido foi inspirado pelo projeto *safe escape* [38] da UNINOVA, e pretende de algum modo contribuir. As principais contribuições são a estrutura proposta em 3.4, a introdução ao simulador *cooja* e análise de uma rede de sensores IoT. A participação no concurso Inncyber Innovation Award [39] na primeira edição foi também fator de motivação. O trabalho proposto foi uma versão adaptada desta tese não tanto na vertente de análise de rede, mas sim numa estrutura de gestão de risco de desastre.

Na primeira parte desta dissertação, em 2, além de um grande trabalho de pesquisa, o foco foi adquirir bases técnicas para dar fundamento ao tema. Foram estudados alguns tópicos, com maior relevância nas redes de sensores sem fios, sistemas ciber-físicos e desastres. Uma vez estudados estes e outros tópicos, houve uma breve discussão de como a tecnologia pode melhorar a gestão de risco de desastre. Tendo em mente algumas lacunas, é feito o enquadramento, donde se conclui que a a tecnologia estudada tem um papel fundamental no ciclo GRD, 2.5.3, e que através desta e de outras metodologias, como a estrutura *Sendai*, pode ter um impacto bastante positivo nos desastres ocorridos nos dias

de hoje. No final deste capítulo, em 2.6.3, é expresso em retrospectiva que o trabalho desenvolvido será focado no plano de evacuação.

No capítulo 3 é feito um planeamento, onde são abordados alguns desafios, é também abordado o ambiente em que a simulação pode eventualmente ser executada, aqui, são apresentadas todas a ferramentas que tornam esta simulação possível, nomeadamente, máquina virtual e simulador, todos os processos são cuidadosamente apresentados, tendo como objetivo dar a conhecer esta ferramenta à equipa da UNINOVA, facilitando eventualmente a sua adoção. Posteriormente, é apresentada a estrutura 3.4 e são definidos os objetivos.

Finalmente, no capítulo 4 são apresentados os detalhes da simulação, a razão da escolha do protocolo de comunicação RPL e uma explicação dos tipos de sensores com o seu código. Posto isto, é feita a primeira simulação, a disposição dos sensores segue a configuração do edifício em ambos os cenários. Na simulação do 1º cenário, os sensores são todos iguais, isto é, registam a mesma temperatura com a mesma frequência, enquanto no segundo cenário existem um conjunto de sensores que regista temperaturas mais elevadas com uma maior frequência.

Após uma análise detalhada da rede, é possível constatar que a maior diferença entre os dois cenários é o consumo energético de ambos, o envio de temperaturas a uma maior frequência requer mais recursos, essencialmente de processamento, comunicação e o tempo que estes sensores se encontram ligados. Uma vez que para uma maior eficiência, estes apenas se ligam quando é realizado o registo e envio de temperatura.

Com a conclusão desta tese, é possível constatar a fiabilidade de uma simulação desta natureza, dentro destas condições o bom desempenho é notório, ainda assim, é importante referir que estamos perante uma simulação, mesmo que altamente fiável, podem existir condicionantes do mundo real, ou qualquer outro imprevisto. Com base em todo o trabalho desenvolvido, e tendo em conta o bom desempenho, seria interessante implementar esta simulação com sensores reais em diferentes ambientes, e estudar vários cenários, apesar dos custos de implementação, certamente iriam surgir um leque de novas possibilidades. Particularmente, existe alguma curiosidade e motivação em implementar a estrutura proposta em 3.4. Além da finalidade principal, uma estrutura que facilita o resgate de pessoas em caso de emergência, existem outras, nomeadamente, controlo de consumo energético do edifício, gestão de ocupação de espaços, aqui é importante ter em atenção às leis de RGPD, além destas metas de um modo geral pretende-se cada vez mais melhorar a experiência de quem frequenta um *smart building*.

Bibliografia

[1] D. Waltenegus e C. Poellabauer, *Fundamentals of Wireless Sensor Networks: Theory and Practice*, Wiley, ed. 2011, pp. 3–7, ISBN: 978-0-470-99765-9. DOI: 10.1002/9780470666388.

[2] J. Fraden, *Handbook of Modern Sensors: Physics, Designs, and Applications*, 5ª ed. Springer, 2016, ISBN: 9783319193021. DOI: 10.1007/978-3-319-19303-8.

[3] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam e E. Cayirci, "Wireless sensor networks: A survey", *Computer Networks*, vol. 38, n.º 4, pp. 393–422, 2002, ISSN: 13891286. DOI: 10.1016/S1389-1286(01)00302-4.

[4] J. Zheng e A. Jamalipour, *Wireless Sensor Networks: A Networking Perspective*. Wiley - IEEE Press, 2009, ISBN: 978-0-470-16763-2.

[5] Statista Research Department, *Internet of Things (IoT) active device connections installed base worldwide from 2015 to 2025*, 2018. URL: https://www.statista.com/statistics/1101442/iot-number-of-connected-devices-worldwide/ (acedido em 19/05/2020).

[6] A. Rayes e S. Salam, *Internet of Things From Hype to Reality*, 2ª ed. Springer, 2019, pp. 1–3, ISBN: 9783319995151. DOI: 10.1007/978-3-319-99516-8.

[7] B. Zhang, X.-X. Ma e Z.-G. Qin, "Security Architecture on the Trusting Internet of Things", *Journal of Electronic Science and Technology*, vol. 9, n.º 4, pp. 364–367, 2011.

[8] R. Alur, *Principles of Cyber-Physical Systems*. London: MIT Press, 2015, pp. 1–2, ISBN: 9780262029117.

[9] E. A. Lee, "Cyber Physical Systems: Design Challenges", EECS Department, University of California, Berkeley, rel. téc., 2008. URL: http://www.eecs.berkeley.edu/Pubs/TechRpts/2008/EECS-2008-8.html.

[10] R. Baheti e H. Gill, "Cyber-Physical Systems", *IEEE Access*, *The Impact of Control Technology*, vol. 3, n.º 6, pp. 821–834, 2011.

[11] K. K. Venkatasubramanian, "Security solutions for cyber-physical systems", tese de doutoramento, Arizona State University, 2009, pp. 2–3.

[12] M. Broy, M. Cengarle e E. Geisberger, "Cyber-Physical Systems: Imminent Challenges", Technische Universitat Munchen e fortiss GmbH, vol. 7539, Springer Berlin Heidelberg, 2012, pp. 1–28. DOI: 10.1007/978-3-642-34059-8_1.

[13] C. Liu e Y. Zhang, *Cyber Physical Systems: Architectures, Protocols and Applications*, 3. CRC Press, 2015, vol. 93, pp. 7–9, ISBN: 9780429160363. DOI: 10.1201/b19003. URL: https://books.google.pt/books?id=2fIYCwAAQBAJ.

[14] R. H. Weber e E. Studer, "Cybersecurity in the Internet of Things: Legal aspects", *Computer Law and Security Review*, vol. 32, n.º 5, pp. 715–728, 2016. DOI: 10.1016/j.clsr.2016.07.002.

[15] A. A. Nazarenko e L. M. Camarinha-Matos, "Towards collaborative Cyber-Physical Systems", em *2017 International Young Engineers Forum (YEF-ECE)*, Caparica: IEEE, 2017, pp. 12–17, ISBN: 978-1-5090-4639-3. DOI: 10.1109/YEF-ECE.2017.7935633.

[16] UNESCO, *Managing Disaster Risks for World Heritage*. Paris: United nations educational, scientific e cultural organization (UNESCO), 2010, ISBN: 978-92-3-104165-5.

[17] *Organização Meteorológica Mundial*. URL: https://severeweather.wmo.int/tc/cgn/acronyms.html (acedido em 25/04/2020).

[18] B. Christofaro, *Cyberattacks are the newest frontier of war and can strike harder than a natural disaster*, 2019. URL: https://www.businessinsider.com/cyber-attack-us-struggle-taken-offline-power-grid-2019-4 (acedido em 26/04/2020).

[19] Marsh & McLennan Companies (MMC), "MMC Cyber Handbook 2019", rel. téc., 2019, p. 66.

[20] D. Alexander, *Principles of Emegency Planning and Management*, 6ª ed. Terra, 2012, p. 6, ISBN: 978-1-903544-10-5.

[21] UNDRR (United Nations Office for Disaster Risk Reduction), "Escritório das Nações Unidas para Redução do Risco de Desastres: Relatório Anual", Genebra, rel. téc., 2019, p. 41.

[22] D. Jiao e J. Sun, "Real-time visualization of geo-sensor data based on the protocol-coupling symbol construction method", *ISPRS International Journal of Geo-Information*, vol. 7, n.º 12, 2018, ISSN: 22209964. DOI: 10.3390/ijgi7120460.

[23] Y. Hong, D. Li, Q. Wu e H. Xu, "Dynamic Route Network Planning Problem for Emergency Evacuation in Restricted-Space Scenarios", *Journal of Advanced Transportation*, vol. 2018, 2018, ISSN: 20423195. DOI: 10.1155/2018/4295419.

[24] M. Min, J. Lee e S. Lim, "Effective evacuation route planning algorithms by updating earliest arrival time of multiple paths", *Proceedings of the 3rd ACM SIGSPATIAL International Workshop on Mobile Geographic Information Systems, MobiGIS 2014 - In Conjunction with the 22nd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, ACM SIGSPATIAL 2014*, pp. 8–17, 2014. DOI: 10.1145/2675316.2675326.

[25] S. Shekhar, K. S. Yang, V. M. Gunturi, L. Manikonda, D. Oliver, X. Zhou, B. George, S. Kim, J. M. Wolff e Q. Lu, "Experiences with evacuation route planning algorithms", *International Journal of Geographical Information Science*, vol. 26, n.º 12, pp. 2253–2265, 2012, ISSN: 13658816. DOI: 10.1080/13658816.2012.719624.

[26] H. Grindvoll, O. Vermesan, T. Crosbie, R. Bahr, N. Dawood e G. M. Revel, "A wireless sensor network for intelligent building energy managementbased on multi communication standards - A case study", *Journal of Information Technology in*, vol. 17, n.º December 2010, pp. 43–62, 2012, ISSN: 1874-4753.

[27] Siemens, "Smart sensors: the roots of building connectivity and intelligence", vol. 2, pp. 1–3, 2019.

[28] T. H. Clausen, J. Yi e Y. Igarashi, "Evaluation of routing protocol for low power and Lossy Networks: LOADng and RPL", em *Conference on Wireless Sensor (ICWISE)*, Kuching, Malaysia, 2013, pp. 19–24. DOI: 10.1109/ICWISE.2013.6728773..

[29] A. V. Krishna e M. K. G, "A Light Weight Secure Protocol for Quick Disaster Recovery using AODV", *International Journal of Engineering Research & Technology (IJERT)*, vol. 8, n.º 06, pp. 1373–1378, 2019.

[30] S. Tyagi, S. Som e Q. P. Rana, "A Reliability based Variant of AODV in MANETs: Proposal, Analysis and Comparison", *Procedia Computer Science*, vol. 79, pp. 903–911, 2016, ISSN: 18770509. DOI: 10.1016/j.procs.2016.03.112.

[31] M. Arraiza, "RDC protocols in Wireless Sensor Networks running Contiki", tese de doutoramento, Universidad Pública de Navarra e Vrije Universiteit Brussel, 2015, pp. 10–11.

[32] Q. Toheed e H. Razi, "Asymmetric-Key Cryptography for Contiki", tese de doutoramento, Chalmers University of Technology e University of Gothenburg, 2010, p. 10.

[33] J. Teixeira, "Wireless Sensor Network for Forest Fire Detection", tese de doutoramento, FEUP, 2017, p. 21.

[34] C. Thomson, I. Romdhani, A. Al-Dubai, M. Qasem, B. Ghaleb e I. Wadhaj, *Cooja Simulator Manual*, 1.0. Edinburgh Napier University, 2016.

[35] K. Zen, D. Habibi, A. Rassau e I. Ahmad, "Performance evaluation of IEEE 802.15.4 for mobile sensor networks", *5th IEEE and IFIP International Conference on Wireless and Optical Communications Networks, WOCN 2008*, 2008. DOI: 10.1109/WOCN.2008.4542536.

[36] B. Al Nahas, "Multichannel Communication in Contiki's low-power IPv6 Stack", p. 47, 2013. URL: http://www.diva-portal.org/smash/record.jsf?pid=diva2:629491&dswid=-4378.

[37] Moteiv Corporation, "Tmote Sky Datasheet", San Francisco, rel. téc., 2006, p. 1.

[38] L. M. Camarinha-matos, A. I. Oliveira, P. Pereira e Javad Jassbi, "Collaborative Safe Escape in Digital Transformation", em *20th IFIP WG 5.5 Working Conference on Virtual Enterprises*, vol. 568, Turin, Italy: Springer, 2019, pp. 431–444, ISBN: 9783030284633. DOI: 10.1007/978-3-030-28464-0. URL: http://dx.doi.org/10.1007/978-3-030-28464-0_1.

[39] Premivalor, Altice e EDP, *INNCYBER Innovation AWARD 2020*, 2020. URL: https://www.inncyberinnovationhub.com/ (acedido em 08/05/2021).

[40] Visual Paradigm, *Fire Evacuation Plan Template*. URL: https://online.visual-paradigm.com/app/diagrams/#diagram:proj=0&type=Flowchart&gallery=/repository/cd749460-0a1b-4f49-b1d7-d9be2a3b045a.xml&name=FireEvacuationPlan (acedido em 05/06/2020).

APÊNDICE A

ANEXO 1

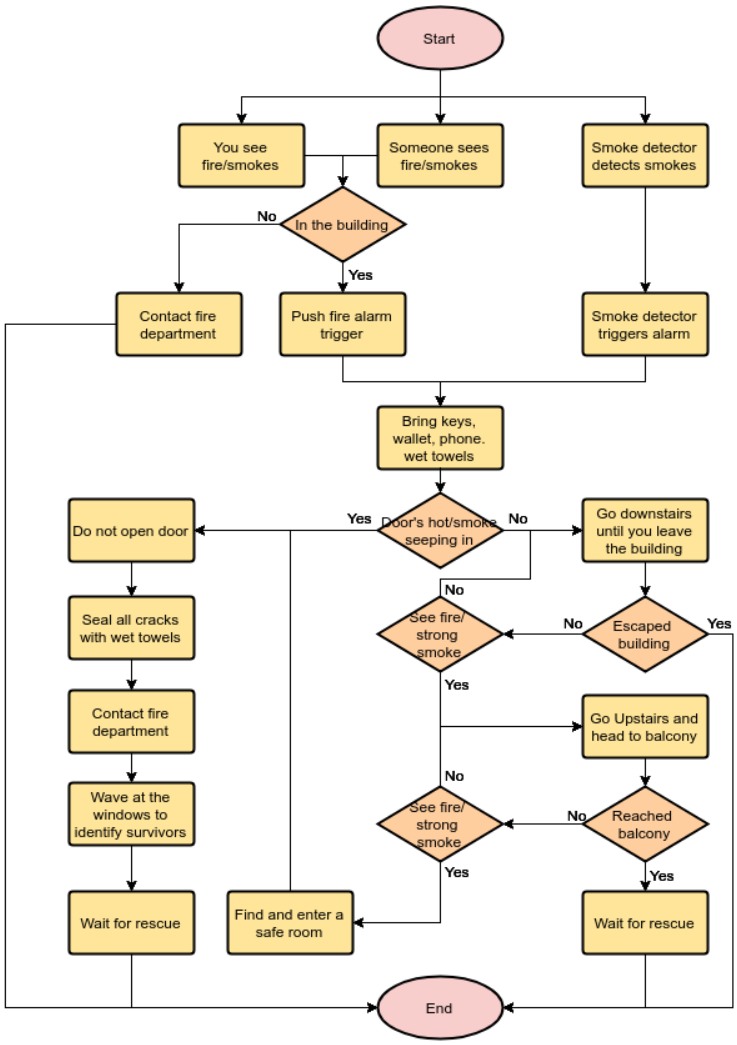

Figura A.1: Diagrama de atividade da reacção a um desastre [40]

APÊNDICE B

ANEXO 2

```
1
2  /*
3   * Redistribution and use in source and binary forms, with or without
4   * modification, are permitted provided that the following conditions
5   * are met:
6   * 1. Redistributions of source code must retain the above copyright
7   *    notice, this list of conditions and the following disclaimer.
8   * 2. Redistributions in binary form must reproduce the above copyright
9   *    notice, this list of conditions and the following disclaimer in the
10  *    documentation and/or other materials provided with the distribution.
11  * 3. Neither the name of the Institute nor the names of its contributors
12  *    may be used to endorse or promote products derived from this software
13  *    without specific prior written permission.
14  *
15  * THIS SOFTWARE IS PROVIDED BY THE INSTITUTE AND CONTRIBUTORS ``AS IS'' AND
16  * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
17  * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
18  * ARE DISCLAIMED.  IN NO EVENT SHALL THE INSTITUTE OR CONTRIBUTORS BE LIABLE
19  * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
20  * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
21  * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
22  * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
23  * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
24  * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
25  * SUCH DAMAGE.
26  *
27  * This file is part of the Contiki operating system.
28  *
29  */
30
31 #include "contiki.h"
```

```
#include "net/ip/uip.h"
#include "net/ipv6/uip-ds6.h"
#include "net/ip/uip-udp-packet.h"
#include "net/rpl/rpl.h"
#include "dev/serial-line.h"
#if CONTIKI_TARGET_Z1
#include "dev/uart0.h"
#else
#include "dev/uart1.h"
#endif
#include "collect-common.h"
#include "collect-view.h"

#include <stdio.h>
#include <string.h>

#define UDP_CLIENT_PORT 8775
#define UDP_SERVER_PORT 5688

#define DEBUG DEBUG_PRINT
#include "net/ip/uip-debug.h"

static struct uip_udp_conn *client_conn;
static uip_ipaddr_t server_ipaddr;

/*---------------------------------------------------------------------------*/
PROCESS(udp_client_process, "UDP client process");
AUTOSTART_PROCESSES(&udp_client_process, &collect_common_process);
/*---------------------------------------------------------------------------*/
void
collect_common_set_sink(void)
{
  /* A udp client can never become sink */
}
/*---------------------------------------------------------------------------*/

void
collect_common_net_print(void)
{
  rpl_dag_t *dag;
  uip_ds6_route_t *r;

  /* Let's suppose we have only one instance */
  dag = rpl_get_any_dag();
  if(dag->preferred_parent != NULL) {
    PRINTF("Preferred parent: ");
    PRINT6ADDR(rpl_get_parent_ipaddr(dag->preferred_parent));
```

```
79      PRINTF("\n");
80    }
81    for(r = uip_ds6_route_head();
82        r != NULL;
83        r = uip_ds6_route_next(r)) {
84      PRINT6ADDR(&r->ipaddr);
85    }
86    PRINTF("---\n");
87  }
88  /*---------------------------------------------------------------------------
      */
89  static void
90  tcpip_handler(void)
91  {
92    if(uip_newdata()) {
93      /* Ignore incoming data */
94    }
95  }
96  /*---------------------------------------------------------------------------
      */
97  void
98  collect_common_send(void)
99  {
100   static uint8_t seqno;
101   struct {
102     uint8_t seqno;
103     uint8_t for_alignment;
104     struct collect_view_data_msg msg;
105   } msg;
106   /* struct collect_neighbor *n; */
107   uint16_t parent_etx;
108   uint16_t rtmetric;
109   uint16_t num_neighbors;
110   uint16_t beacon_interval;
111   rpl_parent_t *preferred_parent;
112   linkaddr_t parent;
113   rpl_dag_t *dag;
114
115   if(client_conn == NULL) {
116     /* Not setup yet */
117     return;
118   }
119   memset(&msg, 0, sizeof(msg));
120   seqno++;
121   if(seqno == 0) {
122     /* Wrap to 128 to identify restarts */
123     seqno = 128;
124   }
125   msg.seqno = seqno;
126
```

```
127   linkaddr_copy(&parent, &linkaddr_null);
128   parent_etx = 0;
129
130   /* Let's suppose we have only one instance */
131   dag = rpl_get_any_dag();
132   if(dag != NULL) {
133     preferred_parent = dag->preferred_parent;
134     if(preferred_parent != NULL) {
135       uip_ds6_nbr_t *nbr;
136       nbr = uip_ds6_nbr_lookup(rpl_get_parent_ipaddr(preferred_parent));
137       if(nbr != NULL) {
138         /* Use parts of the IPv6 address as the parent address, in reversed
      byte order. */
139         parent.u8[LINKADDR_SIZE - 1] = nbr->ipaddr.u8[sizeof(uip_ipaddr_t) -
      2];
140         parent.u8[LINKADDR_SIZE - 2] = nbr->ipaddr.u8[sizeof(uip_ipaddr_t) -
      1];
141         parent_etx = rpl_get_parent_rank((uip_lladdr_t *) uip_ds6_nbr_get_ll(
      nbr)) / 2;
142       }
143     }
144     rtmetric = dag->rank;
145     beacon_interval = (uint16_t) ((2L << dag->instance->dio_intcurrent) /
      1000);
146     num_neighbors = uip_ds6_nbr_num();
147   } else {
148     rtmetric = 0;
149     beacon_interval = 0;
150     num_neighbors = 0;
151   }
152
153   /* num_neighbors = collect_neighbor_list_num(&tc.neighbor_list); */
154   collect_view_construct_message(&msg.msg, &parent,
155                                  parent_etx, rtmetric,
156                                  num_neighbors, beacon_interval);
157
158   uip_udp_packet_sendto(client_conn, &msg, sizeof(msg),
159                         &server_ipaddr, UIP_HTONS(UDP_SERVER_PORT));
160 }
161 /*---------------------------------------------------------------------------
      */
162 void
163 collect_common_net_init(void)
164 {
165 #if CONTIKI_TARGET_Z1
166   uart0_set_input(serial_line_input_byte);
167 #else
168   uart1_set_input(serial_line_input_byte);
169 #endif
170   serial_line_init();
```

```
171 }
172 /*---------------------------------------------------------------------------
    */
173 static void
174 print_local_addresses(void)
175 {
176   int i;
177   uint8_t state;
178
179   PRINTF("Client IPv6 addresses: ");
180   for(i = 0; i < UIP_DS6_ADDR_NB; i++) {
181     state = uip_ds6_if.addr_list[i].state;
182     if(uip_ds6_if.addr_list[i].isused &&
183        (state == ADDR_TENTATIVE || state == ADDR_PREFERRED)) {
184       PRINT6ADDR(&uip_ds6_if.addr_list[i].ipaddr);
185       PRINTF("\n");
186       /* hack to make address "final" */
187       if (state == ADDR_TENTATIVE) {
188         uip_ds6_if.addr_list[i].state = ADDR_PREFERRED;
189       }
190     }
191   }
192 }
193 /*---------------------------------------------------------------------------
    */
194 static void
195 set_global_address(void)
196 {
197   uip_ipaddr_t ipaddr;
198
199   uip_ip6addr(&ipaddr, UIP_DS6_DEFAULT_PREFIX, 0, 0, 0, 0, 0, 0, 0);
200   uip_ds6_set_addr_iid(&ipaddr, &uip_lladdr);
201   uip_ds6_addr_add(&ipaddr, 0, ADDR_AUTOCONF);
202
203   /* set server address */
204   uip_ip6addr(&server_ipaddr, UIP_DS6_DEFAULT_PREFIX, 0, 0, 0, 0, 0, 0, 1);
205
206 }
207 /*---------------------------------------------------------------------------
    */
208 PROCESS_THREAD(udp_client_process, ev, data)
209 {
210   PROCESS_BEGIN();
211
212   PROCESS_PAUSE();
213
214   set_global_address();
215
216   PRINTF("UDP client process started\n");
217
```

```
218   print_local_addresses();
219
220   /* new connection with remote host */
221   client_conn = udp_new(NULL, UIP_HTONS(UDP_SERVER_PORT), NULL);
222   udp_bind(client_conn, UIP_HTONS(UDP_CLIENT_PORT));
223
224   PRINTF("Created a connection with the server ");
225   PRINT6ADDR(&client_conn->ripaddr);
226   PRINTF(" local/remote port %u/%u\n",
227         UIP_HTONS(client_conn->lport), UIP_HTONS(client_conn->rport));
228
229   while(1) {
230     PROCESS_YIELD();
231     if(ev == tcpip_event) {
232       tcpip_handler();
233     }
234   }
235
236   PROCESS_END();
237 }
238 /*---------------------------------------------------------------------------
      */
```

Listagem B.1: Código completo de udp-sender.c

Apêndice C

Anexo 3

```
/*
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 * 3. Neither the name of the Institute nor the names of its contributors
 *    may be used to endorse or promote products derived from this software
 *    without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE INSTITUTE AND CONTRIBUTORS ``AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED.  IN NO EVENT SHALL THE INSTITUTE OR CONTRIBUTORS BE LIABLE
 * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
 * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
 * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
 * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
 * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
 * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 *
 * This file is part of the Contiki operating system.
 *
 */

#include "contiki.h"
#include "contiki-lib.h"
```

```
32 #include "contiki-net.h"
33 #include "net/ip/uip.h"
34 #include "net/rpl/rpl.h"
35 #include "net/linkaddr.h"
36
37 #include "net/netstack.h"
38 #include "dev/button-sensor.h"
39 #include "dev/serial-line.h"
40 #if CONTIKI_TARGET_Z1
41 #include "dev/uart0.h"
42 #else
43 #include "dev/uart1.h"
44 #endif
45 #include <stdio.h>
46 #include <stdlib.h>
47 #include <string.h>
48 #include <ctype.h>
49 #include "collect-common.h"
50 #include "collect-view.h"
51
52 #define DEBUG DEBUG_PRINT
53 #include "net/ip/uip-debug.h"
54
55 #define UIP_IP_BUF   ((struct uip_ip_hdr *)&uip_buf[UIP_LLH_LEN])
56
57 #define UDP_CLIENT_PORT 8775
58 #define UDP_SERVER_PORT 5688
59
60 static struct uip_udp_conn *server_conn;
61
62 PROCESS(udp_server_process, "UDP server process");
63 AUTOSTART_PROCESSES(&udp_server_process,&collect_common_process);
64 /*---------------------------------------------------------------------------
    */
65 void
66 collect_common_set_sink(void)
67 {
68 }
69 /*---------------------------------------------------------------------------
    */
70 void
71 collect_common_net_print(void)
72 {
73   printf("I am sink!\n");
74 }
75 /*---------------------------------------------------------------------------
    */
76 void
77 collect_common_send(void)
78 {
```

```
79    /* Server never sends */
80  }
81  /*---------------------------------------------------------------------------
        */
82  void
83  collect_common_net_init(void)
84  {
85  #if CONTIKI_TARGET_Z1
86    uart0_set_input(serial_line_input_byte);
87  #else
88    uart1_set_input(serial_line_input_byte);
89  #endif
90    serial_line_init();
91
92    PRINTF("I am sink!\n");
93  }
94  /*---------------------------------------------------------------------------
        */
95  static void
96  tcpip_handler(void)
97  {
98    uint8_t *appdata;
99    linkaddr_t sender;
100   uint8_t seqno;
101   uint8_t hops;
102
103   if(uip_newdata()) {
104     appdata = (uint8_t *)uip_appdata;
105     sender.u8[0] = UIP_IP_BUF->srcipaddr.u8[15];
106     sender.u8[1] = UIP_IP_BUF->srcipaddr.u8[14];
107     seqno = *appdata;
108     hops = uip_ds6_if.cur_hop_limit - UIP_IP_BUF->ttl + 1;
109     collect_common_recv(&sender, seqno, hops,
110                         appdata + 2, uip_datalen() - 2);
111   }
112 }
113 /*---------------------------------------------------------------------------
        */
114 static void
115 print_local_addresses(void)
116 {
117   int i;
118   uint8_t state;
119
120   PRINTF("Server IPv6 addresses: ");
121   for(i = 0; i < UIP_DS6_ADDR_NB; i++) {
122     state = uip_ds6_if.addr_list[i].state;
123     if(state == ADDR_TENTATIVE || state == ADDR_PREFERRED) {
124       PRINT6ADDR(&uip_ds6_if.addr_list[i].ipaddr);
125       PRINTF("\n");
```

```
126       /* hack to make address "final" */
127       if (state == ADDR_TENTATIVE) {
128         uip_ds6_if.addr_list[i].state = ADDR_PREFERRED;
129       }
130     }
131   }
132 }
133 /*---------------------------------------------------------------------------
      */
134 PROCESS_THREAD(udp_server_process, ev, data)
135 {
136   uip_ipaddr_t ipaddr;
137   struct uip_ds6_addr *root_if;
138
139   PROCESS_BEGIN();
140
141   PROCESS_PAUSE();
142
143   SENSORS_ACTIVATE(button_sensor);
144
145   PRINTF("UDP server started\n");
146
147 #if UIP_CONF_ROUTER
148   uip_ip6addr(&ipaddr, UIP_DS6_DEFAULT_PREFIX, 0, 0, 0, 0, 0, 0, 1);
149   /* uip_ds6_set_addr_iid(&ipaddr, &uip_lladdr); */
150   uip_ds6_addr_add(&ipaddr, 0, ADDR_MANUAL);
151   root_if = uip_ds6_addr_lookup(&ipaddr);
152   if(root_if != NULL) {
153     rpl_dag_t *dag;
154     dag = rpl_set_root(RPL_DEFAULT_INSTANCE,(uip_ip6addr_t *)&ipaddr);
155     uip_ip6addr(&ipaddr, UIP_DS6_DEFAULT_PREFIX, 0, 0, 0, 0, 0, 0, 0);
156     rpl_set_prefix(dag, &ipaddr, 64);
157     PRINTF("created a new RPL dag\n");
158   } else {
159     PRINTF("failed to create a new RPL DAG\n");
160   }
161 #endif /* UIP_CONF_ROUTER */
162
163   print_local_addresses();
164
165   /* The data sink runs with a 100% duty cycle in order to ensure high
166      packet reception rates. */
167   NETSTACK_RDC.off(1);
168
169   server_conn = udp_new(NULL, UIP_HTONS(UDP_CLIENT_PORT), NULL);
170   udp_bind(server_conn, UIP_HTONS(UDP_SERVER_PORT));
171
172   PRINTF("Created a server connection with remote address ");
173   PRINT6ADDR(&server_conn->ripaddr);
174   PRINTF(" local/remote port %u/%u\n", UIP_HTONS(server_conn->lport),
```

```
175           UIP_HTONS(server_conn->rport));
176
177   while(1) {
178     PROCESS_YIELD();
179     if(ev == tcpip_event) {
180       tcpip_handler();
181     } else if (ev == sensors_event && data == &button_sensor) {
182       PRINTF("Initiating global repair\n");
183       rpl_repair_root(RPL_DEFAULT_INSTANCE);
184     }
185   }
186
187   PROCESS_END();
188 }
189 /*---------------------------------------------------------------------------
      */
190
191
192
```

Listagem C.1: Código completo de udp-sink.c