Jorge Manuel Marques Gonçalves

Sistema de Comunicação Resistente a Falhas entre Sensores

Departamento de Ciência de Computadores
Faculdade de Ciências da Universidade do Porto
Setembro de 2012

Jorge Manuel Marques Gonçalves

Sistema de Comunicação Resistente a Falhas entre Sensores

Dissertação submetida à Faculdade de Ciências da
Universidade do Porto como parte dos requisitos para a obtenção do grau de
Mestre em Engenharia de Redes e Sistemas Informáticos

Orientador: Roberto Ugo Di Cera Colazingari
Co-orientador: Rui Pedro de Magalhães Claro Prior

Departamento de Ciência de Computadores
Faculdade de Ciências da Universidade do Porto
Setembro de 2012

Agradecimentos

É com muita satisfação que expresso os meus mais sinceros agradecimentos a todas as pessoas que me acompanharam durante este estágio, e que, de alguma forma, tornaram possível a realização deste projecto.

Assim, começo por agradecer à empresa *Around Knowledge* pela hospitalidade e disponibilidade sempre demonstrada para a concretização do projecto. Deixo uma palavra de especial agradecimento ao meu orientador de estágio, Roberto Colazingari, à Suzy Vasconcelos, à Diana Almeida, e de uma forma geral, a todos os que me proporcionaram um ambiente de trabalho saudável.

Agradeço também a total disponibilidade, confiança, compreensão e sabedoria transmitida pelo meu co-orientador, Rui Prior, e a todos os professores da Faculdade de Ciências da Universidade do Porto, que de alguma forma, contribuíram para o meu crescimento enquanto profissional.

Não podia deixar de agradecer aos meus pais, Manuel Gonçalves e Maria de Lurdes Marques, e irmão, Tiago Gonçalves, pelo carinho e motivação sempre prestada. Um muito obrigada à minha namorada, Tânia Pinho, por toda a dedicação, compreensão, motivação e ajuda.

Por fim, sem menos importância, um agradecimento sincero aos meus amigos João Teixeira e Luís Sardinha, pela amizade, companheirismo e interajuda ao longo de todo este processo académico. A todos, um muito obrigado!

"Para cultivar a sabedoria, é preciso força interior. Sem crescimento interno, é difícil conquistar a autoconfiança e a coragem necessárias. Sem elas, a nossa vida complica-se. O impossível torna-se possível com a força de vontade."

Dalai Lama.

Resumo

Vivemos numa sociedade cada vez mais dependente das novas tecnologias e dos benefícios que elas nos oferecem. As redes de distribuição eléctrica actuais caminham a passos largos para o limite da sua exploração, havendo necessidade de procurar novas alternativas que permitam potenciar o seu uso. Assim, as *smart grids* são vistas como uma solução, de forma a controlar e gerir a rede em tempo real.

A inclusão das *smart grids* contém ainda uma série de desafios e foi neste sentido que foi aceite o desafio proposto por um fornecedor de energia português, que consiste em analisar e corrigir algumas falhas na rede de sensores, nomeadamente a nível de segurança e fiabilidade da comunicação. Para atender a todas as necessidades do fornecedor, foi inicialmente feita uma análise de requisitos para a solução a apresentar.

Foi iniciado um estudo acerca do *hardware* a utilizar na solução final, assim como, em que medida as características da pilha protocolar do ZigBee nos poderia ajudar a solucionar alguns dos problemas apresentados. Para além disso, foram realizados alguns testes à componente rádio do *hardware* escolhido, no qual se pretende analisar as várias potencialidades e limitações.

Por fim, foi construído um protótipo para a rede de sensores, que visa automatizar a gestão e monitorização da mesma, contemplando mecanismos de fiabilidade e de segurança na troca de mensagens, mecanismos de alerta de anomalias e efectuar operações remotas na rede. Este protótipo permite aumentar a independência da rede, melhorar a qualidade de serviço, diminuir os custos de operação e potenciar a eficiência energética.

Abstract

We live in a society that reflects an increasing dependence on new technologies and the benefits they provide. Electrical distribution networks nowadays are taking long steps towards the limit of their operations, ensuring there is a need to look for new alternatives to enhance their use. Smart grids constitute a solution to this problem, as they provide a way of controlling and managing a grid in real-time.

The inclusion of smart grids still presents a number of challenges and is in this sense that the challenge proposed by a Portuguese energy provider was accepted, the objective of the proposal was analyzing and fixing some flaws in the sensor network, and in particular at the level of security and reliability of the communication. In order to meet all of the provider's needs, a requirements analysis was first performed.

The hardware to be used in the final solution was then investigated. Issues such as how the characteristics of ZigBee network protocol layer could help in solving the problems at hand, were also considered. Some tests were undertaken on the radio component of the chosen hardware, in order to evaluate its capabilities and limitations.

Finally, a prototype was built into the sensor grid, that aims at automating it's managing and monitoring by accommodating security and reliability mechanisms in the exchange of messages. It also features anomaly reporting mechanisms as well as the ability of performing remote operations in the grid. This prototype allows the increase of the grid's independence, the improvement of the quality of service, the decrease of maintenance costs and a higher energetic efficiency.

Conteúdo

Lista de Tabelas

Lista de Figuras

Acrónimos

ACK Acknowledgment

AES Advanced Encryption Standard

API Application Programming Interface

AF Application Framework

APL Application Layer

APS Application Support Sublayer

ARM Advanced RISC Machine

BPSK Binary Phase-Shift Keying

DL Data Logger

DNS Domain Name System

DV Distance Vector

FFD Full-Function Device

IEEE Institute of Electrical and Electronics Engineers

IP Internet Protocol

JTAG Joint Test Action Group

JSON JavaScript Object Notation

HTTPS Hypertext Transfer Protocol Secure

LTS Long-Term Support

MAC Medium Access Control

MB Megabyte

MIC Message Integrity Code

NWK Network Layer

O-QPSK Offset Quadrature Phase-Shift Keying

PAN Personal Area Network

PHY Physical Layer

RAM Random-Access Memory

RF Radio-Frequency

RFD Reduced-Function Device

RSSI Received Signal Strength Indicator

RTT Round Trip Time

SKKE Symmetric-Key Key Exchange

SQL Structured Query Language

WLAN Wireless Local Area Network

WSN Wireless Sensor Networks

ZC ZigBee Coordinator

ZDO ZigBee Device Object

ZED ZigBee End Device

ZR ZigBee Router

Este relatório de estágio não foi escrito ao abrigo do novo Acordo Ortográfico.

Capítulo 1

Introdução

Associado ao crescimento de equipamentos electrónicos capazes de comunicarem uns com os outros, o mercado das comunicações tem sofrido um elevado crescimento nas últimas décadas. Uma das áreas que tem contribuído para este crescimento são as comunicações sem fio.

As comunicações sem fios têm-se popularizado pela sua facilidade de instalação, evitando o uso de cabos para ligar os equipamentos que, por consequência, diminui o custo e aumenta a escalabilidade e a flexibilidade dos equipamentos da rede.

Os avanços na micro-electrónica e nas redes sem fio permitiram a criação de redes com equipamentos simples agregados a sensores de recolha, as quais costumamos designar por redes de sensores (Wireless Sensor Networks (WSN)). Propriedades como dimensão reduzida, baixo consumo e baixo custo têm contribuído para o crescimento deste tipo de redes, comparativamente a outras redes sem fios (Wireless Local Area Network (WLAN), Bluetooth, ...).

Outro aspecto, que é cada vez mais tido em conta em redes de comunicações, é a garantia da qualidade e o desempenho dos serviços por ela fornecidos. A monitorização por parte dos administradores torna-se a cada dia que passa mais complexa, morosa e entediante com este crescimento. Automatizar o processo de monitorização permite detectar anomalias no momento, gerar alertas, solucionar problemas regulares de forma automática e minimizar custos.

1.1 Contextualização

As redes eléctricas actuais caminham para o limite da sua exploração. O aumento da procura pela electricidade exige uma maior eficiência, o que, de dia para dia, faz crescer o número de desafios associados aos compromissos ambientais. Por essa razão, as redes eléctricas estão a ganhar inteligência e a transformar-se em *smart grids*.

O sistema actual de distribuição de rede não está preparado para lidar com as oscilações provenientes do aumento de dispositivos ligados à corrente, criando um aumento no consumo de energia.

As *smart grids* podem ser vistas como uma solução, pois permitem uma monitorização adequada de toda a rede, necessária para garantir a troca de informação em tempo real, através do uso de sensores que assegura essa mesma informação. A rede eléctrica torna-se mais flexível, inteligente, controlável e protegida do que as redes actuais.

Uma grande vantagem sob o ponto de vista do consumidor, é o facto de poder controlar os seus próprios gastos, dando por isso a este, um papel activo em termos de eficiência energética na própria habitação. Também possibilitam políticas de preços dinâmicos que incentivem o uso quando a rede tem menos carga.

1.2 Apresentação da Around Knowledge

A Around Knowledge [17] - Consultadoria Informática Lda (AK) é uma empresa sediada e incubada em INSerralves, na cidade do Porto. A AK foi criada em Maio de 2009, por três investigadores universitários que têm como principal objectivo estabelecer uma ponte entre o mundo académico e o empresarial.

Segundo o INE, a empresa está intimamente ligada a actividades de programação, incluindo as seguintes: "actividades de concepção, desenvolvimento, modificação, teste e assistência a programas informáticos (*software*) de acordo com a necessidade de um cliente específico. Inclui programação de sistemas, aplicações, base de dados e páginas *web*".

Desde a sua criação, a AK tem conquistado vários prémios de elevada importância, nomeadamente, o *"Projecto BIPS"*, *"Desafio Securitas Systems - Novos Talentos - Inovação em Segurança Electrónica"*, *"Sapo Summerbits"*, *"GSI - Accelerators Startup Challenge 2011"*, entre outros. É ainda de salientar a ligação que a empresa possui com

o INESC TEC - Laboratório Associado, mais concretamente com o centro CRACS na Faculdade de Ciências da Universidade do Porto.

O projecto BIPS pode ser considerado o impulsionador da AK, uma vez que o seu crescimento surgiu desde então, com a conquista do prémio promovido pelo ISCTE em parceria com o MIT. O BIPS surge como uma ideia inovadora capaz de perceber em tempo real, através de um sistema de posicionamento em espaços interiores e exteriores por radiofrequências, tempos e rotas nas deslocações de indivíduos. O BIPS está a ser patenteado internacionalmente.

Relativamente ao *"GSI - Accelerators Startup Challenge 2011"*, este prémio deu acesso a um plano de desenvolvimento de negócios durante o período de três meses na PPTC - *Play and Plug Tech Center* em *Silicon Valley*.

Por fim, AK decidiu alargar a área de actuação de mercado, fazendo uso das suas competências e conhecimentos científicos e técnicos para o desenvolvimento de aplicações móveis, em diversas áreas de negócio, tanto em território nacional como internacional.

1.3 Motivação

Este projecto surgiu com a necessidade de criar um processo de controlo automatizado, optimizar a distribuição de energia e facturar a energia consumida. O objectivo, proposto por um fornecedor de energia português, é recolher informações sobre o consumo dos clientes num determinado instante. A longo prazo, a empresa pretende saber onde e quando é necessário disponibilizar mais ou menos energia, de forma a rentabilizar os gastos e a melhorar a qualidade de serviço. Esta ideia será ainda aplicada aos consumos de água e de gás.

Esse fornecedor, lançou um concurso nacional, no intuito de analisar e ponderar a melhor ideia apresentada. Finalizado este, foi escolhida a ideia que lhe pareceu mais adequada à situação. No entanto, após a sua implementação e já durante os testes realizados em lares-piloto, esta veio-se a tornar inadequada para as suas pretensões.

Surgiu então a ideia de melhorar a rede de sensores, que é responsável por recolher dados relativamente aos gastos de água, gás, electricidade, temperatura e a humidade de vários estabelecimentos, nomeadamente, casas, prédios e empresas.

Numa primeira fase, estes sensores permitirão fornecer as contagens dos contadores clássicos, evitando a deslocação de técnicos aos estabelecimentos, de forma a registar

todos os dados necessários. Posteriormente, a ideia será expandida aos automóveis eléctricos.

Com a devida estruturação e construção desta nova rede, será possível a criação de um portal, onde permitirá monitorizar os sensores e os seus gastos associados. Assim, qualquer configuração ou alteração remota será realizada sem qualquer deslocação por parte dos técnicos.

Concluído este projecto, o fornecedor de energia poderá distribuir a sua energia de forma mais eficiente e disponibilizar novos serviços ao cliente, baseando-se na recolha e tratamento dos dados recolhidos, de forma a obter maior satisfação por dos seus clientes.

1.4 Objectivos

O sistema actualmente instalado, não satisfaz totalmente as necessidades da empresa responsável, uma vez que contém falhas a nível de entrega e fiabilidade das mensagens.

Os dados gerados pelos sensores de recolha são obtidos periodicamente independentemente do tipo do sensor ou aplicação, realizando operações desnecessárias que, por consequência, levam à utilização de mais recursos computacionais e de rede do que os estritamente necessários.

A comunicação entre os sensores e o agregador[1] é feita através de portadoras rádio. Como é do conhecimento geral, este tipo de comunicações, designadas por comunicações sem fios, estão dependentes do meio em que se encontram. Por este motivo, existe um conjunto de factores que influenciam a entrega das mensagens, fazendo com que por vezes estas sejam perdidas durante a transferência. O sistema actual não contempla quaisquer normas ou medidas com o intuito de minimizar este problema.

Outra falha conhecida prende-se ao facto dos sensores não se identificarem nem autenticarem perante o agregador, permitindo que qualquer outro sensor estabeleça ligação com este e envie dados não fidedignos. Este, como não tem a responsabilidade de efectuar esse controlo, considera-os válidos, o que pode conduzir à introdução de contagens erradas no sistema.

A fim de corrigir estas e outras falhas que serão apresentadas com mais detalhe mais

[1]Equipamento electrónico responsável por estabelecer a ligação entre a empresa que está a prestar o serviço e a habitação ou o estabelecimento que o utiliza.

adiante, os principais objectivos para este projecto são:

- ***Objectivo 1:*** Garantir fiabilidade e disponibilidade da rede na troca de mensagens.
- ***Objectivo 2:*** Garantir mecanismos de segurança na rede, como confidencialidade e integridade dos sensores e das mensagens. A inclusão de novos sensores na rede deve ser controlada e a obtenção do conteúdo das mensagens por parte de utilizadores desconhecidos não deve ser possível.
- ***Objectivo 3:*** Criar mecanismos para gerir e monitorizar a rede de sensores, responsável pelos sensores de recolha instalados. Deve ser possível detectar se os sensores estão disponíveis, saber a taxa de transmissão de mensagens associada a cada sensor, e ainda detectar se os valores recolhidos pelos sensores estão dentro de um limiar estipulado.
- ***Objectivo 4:*** Criar mecanismos de alerta de anomalias na rede que permitam libertas os administradores de operações de gestão e monitorização, tornando-a mais independente.
- ***Objectivo 5:*** Visualizar e alterar parâmetros de rede remotamente. Este objectivo permitirá maior controlo por parte dos administradores da rede, bem como a realização de operações sem necessidade de deslocamento às instalações do cliente.

1.5 Organização do Relatório

Relativamente à estrutura/organização do presente relatório, este encontra-se dividido em 7 capítulos, sendo que cada capítulo está organizado em sub-capítulos.

O capitulo 2 diz respeito à análise do problema, fazendo por isso uma breve explicação sobre redes de sensores, quais as propriedades do sistema actual e os requisitos da solução que será apresentada.

O capitulo 3, intitula-se por Revisão Tecnológica, descrevendo o estado de arte, o *hardware* ponderado para a solução e de que forma será feita a comunicação na rede de sensores, solucionando desde já alguns dos problemas apresentados. Temos ainda a apresentação de alguns métodos que podem minimizar falhas e, portanto, aumentar a fiabilidade da rede.

O capitulo 4, apresenta experiências realizadas ao nível de rede durante a realização do presente trabalho.

Nos capítulos 5 e 6 é apresentada a solução proposta. No capítulo 5 são descritas as tecnologias necessárias, a arquitectura e a visão geral do sistema. No capitulo 6, são apresentados os pormenores técnicos do desenvolvimento do sistema.

Finalmente, no capitulo 7, apresentam-se as conclusões finais deste trabalho, bem como possíveis caminhos para e futura evolução do projecto.

Capítulo 2

Análise do Problema

O propósito deste projecto surgiu com a necessidade de reestruturar a rede de sensores já implementada, através da correcção de algumas falhas, nomeadamente ao nível da entrega e fiabilidade de mensagens.

De forma a analisar esse mesmo problema, inicialmente é feita uma breve introdução sobre redes de sensores, para nos enquadrar no problema e posteriormente é descrita a arquitectura do sistema actual.

Por fim, depois de compreendida a arquitectura, é feita uma análise da mesma identificando todos os requisitos da solução proposta.

2.1 Breve Introdução às Redes de Sensores

Uma WSN consiste num agregado de pequenos dispositivos autónomos, designados por sensores, distribuídos espacialmente, que realizam uma função de monitorização específica e enviam os dados recolhidos para um nó central. As redes mais recentes permitem bi-direccionalidade, pelo que é também possível controlar a actividade do sensor.

O recurso a redes de sensores tem-se tornado cada vez mais comum. Um dos factores que mais tem contribuído para o seu crescimento é o baixo custo dos dispositivos utilizados. Normalmente, estes possuem uma unidade de processamento (microcontrolador), de armazenamento, de memória, de comunicação e ainda uma pequena bateria.

Esta secção tem como objectivo fundamental introduzir as WSNs, destacando os aspectos mais relevantes, com o intuito de fornecer alguns conhecimentos ao leitor sobre este tipo de redes.

2.1.1 Aplicações

Estes dispositivos começaram por ser inicialmente populares com a recolha de informações relativamente à temperatura, humidade, infravermelhos, acústica, vibrações (por exemplo, detectar actividade sísmica), pressão, entre outros. Rapidamente a sua utilização tornou-se comum em diversas áreas, como por exemplo a nível de detecção de desastres, controlo do ambiente e mapas de diversidade, domótica, agricultura, logística e trânsito [16].

A **detecção de desastres ambientais**, nomeadamente incêndios e sismos, pode ser conseguida através da monitorização de locais considerados perigosos, como é o caso de habitações, florestas ou rios. Um exemplo será a recolha do nível e caudal da água num rio para prevenir eventuais cheias [16].

Outra aplicação possível é o **controlo do ambiente e mapas de diversidade**. Verificar o nível de erosão da água numa praia ou recolher o número de animais que vivem numa determinada região são possíveis exemplos [16].

Na área da **domótica** tem-se assistido a um grande investimento por parte das empresas. Numa habitação, normalmente presencia-se um grande desperdício de recursos, tais como, o uso incorrecto do ar condicionado ou lâmpadas a consumir energia eléctrica desnecessária. A utilização dos sensores permite optimizar a utilização destes recursos, minimizando o desperdício [16].

Na **agricultura**, a utilização de uma WSN permite aplicar com precisão a fertilização e o sistema de rega nos produtos, consoante as condições atmosféricas no local [16].

Em algumas **aplicações logísticas**, é vantajoso equipar mercadorias com sensores para permitir o seu acompanhamento durante o transporte ou controlar o *stock* nas lojas ou armazéns [16].

A sua expansão a nível de **trânsito** tem o objectivo de monitorizar ruas para tentar perceber se existe congestionamento. Com este método, outros carros podem ser avisados e seguirem o seu destino numa rota alternativa [16].

2.1.2 Desafios

Apesar das inúmeras aplicações das WSNs, estas têm uma série de desafios, o que naturalmente levou à sua exploração por diversos investigadores, por forma a serem melhoradas.

A **qualidade de serviço** é um grande desafio, uma vez que o tipo de serviço está relacionado com a qualidade do mesmo. Neste caso, factores como atraso e largura de banda são extremamente importantes. A transferência dos pacotes deverá ser feita o mais breve possível e ocupar pouca largura de banda nas aplicações em tempo real. Noutro tipo de aplicações, que não requerem uma resposta rápida, o atraso torna-se menos importante [16].

Um outro desafio é o facto da rede ser **tolerante a falhas**. Nos casos em que o nó se encontra em modo poupança, danificado ou a comunicação entre os dois nós está permanentemente a ser interrompida é de extrema importância que a WSN contenha mecanismos para evitar ou minimizar as consequências deste problema. Neste sentido, a redundância é necessária usando mais nós do que os estritamente necessários ao funcionamento em condições normais para que a rede funcione correctamente em situações de falha [16].

O **tempo de vida**, devido ao facto de em muitos cenários não ser possível alimentar os nós a partir da rede eléctrica, pode também ser considerado um desafio. Nestes casos, é vulgar recorrer-se ao uso de baterias. No entanto, estas têm um determinado tempo de vida e não é viável substituir ou recarrega-las periodicamente. O tempo de vida está directamente relacionado com a qualidade de serviço. Temos então um compromisso: maior investimento energético aumenta a qualidade de serviço, mas diminui o tempo de vida [16].

Normalmente, com o passar do tempo, as redes têm tendência para se tornarem mais complexas com o acréscimo de novos nós. Manter um funcionamento adequado mesmo com o acréscimo de nós e o consequente aumento da complexidade, ao que normalmente lhe chamamos de **escalabilidade**, torna-se num grande desafio nas redes de comunicação actuais. A arquitectura e o protocolo de rede usado têm que ser capazes de suportar esta expansão [16].

Por fim, a **densidade da rede**, que consiste no número de nós por unidade de área, é variável consoante os diferentes cenários e uma má disposição dos nós pode influenciar o desempenho da rede. É necessário ter especial atenção à forma como vão ser colocados espacialmente os nós [16].

Todos estes desafios estão intimamente ligados às *smart grids*, pelo que, devemos estar atentos aos problemas que estes nos poderão causar.

2.1.3 Segurança

Em qualquer tipo de comunicação, os mecanismos de segurança são indispensáveis. As redes de sensores não são excepção à regra. As metas a atingir são idênticas a outro tipo de redes, só que estas têm a particularidade de serem constituídas por dispositivos de baixo custo e onde, normalmente, os recursos computacionais são reduzidos, havendo que ter especial atenção a este factor.

A **recolha de dados passiva** acontece quando o nó intruso, através da sua antena de captação, intercepta os pacotes da rede, obtendo o conteúdo das mensagens enviadas (as mensagens não se encontram cifradas) [30].

Um outro ataque vulgar é a **subversão do nó**, que consiste em controlar ilegitimamente um nó que tem acesso legítimo à rede, obtendo, por exemplo, as chaves de segurança e comprometendo toda a rede de sensores [30].

A corrupção de mensagens é verificada quando um atacante modifica o conteúdo de uma mensagem, comprometendo a sua integridade [30].

A existência de **nós falsos e dados maliciosos** verifica-se em casos em que a rede não contém nenhum método de segurança, sendo que o nó entra na rede insegura, podendo alterar, falsificar, reproduzir ou fazer desaparecer mensagens da rede [30].

Quando um nó adquire a identidade de outros nós legítimos da rede e a usa para fins ilícitos, estamos perante um ***sybil attack***. Este tipo de ataque pode ser usado contra algoritmos de encaminhamento ou topologias de rede. Um caso especial deste ataque pode ocorrer em redes com encaminhamento geográfico onde um nó *"sybil"* pode aparecer em múltiplos locais em simultâneo [30].

Os mecanismos de segurança são factores muito importantes nos sistemas de informação, no entanto, a garantia total da sua protecção nem sempre é possível. Na secção 2.3.2, temos os problemas de segurança que pretendemos ver resolvidos finalizado este projecto e ainda descrito de forma o sistema pode ser afectado caso os mecanismos de segurança não sejam tomados em conta.

2.2 Sistema Actual

Os contadores clássicos de electricidade e gás encontram-se vulgarmente nas habitações em Portugal e são dispositivos demasiado simples, que permitem apenas medir a quantidade de energia eléctrica e gás consumida por cada habitação.

Com o passar dos tempos, surgiu a necessidade de efectuar leituras em tempo real ou quase real, de recolher informações acerca de falhas de energia (vulgarmente chamados por apagões) e monitorizar a qualidade de energia fornecida.

O abandono destes contadores é cada vez mais frequente, porque com a utilização de dispositivos mais sofisticados obtêm-se mais dados para ajudar a gerir a rede e, por consequência, adaptar a oferta comercial, introduzindo, como exemplo, diferentes tarifas de consumo, dependendo do período do dia ou da época do ano. O cliente, por sua vez, fica mais consciente dos seus gastos, podendo gerir e controlá-los a seu belo prazer.

Nesta secção, é apresentado o sistema que se encontra em fase de testes. A inclusão destes novos equipamentos numa primeira fase será feita em simultâneo com os contadores clássicos, com o intuito de verificar se é viável a utilização desta tecnologia.

A WSN contém dois tipos de equipamentos: dispositivos simples que estarão espalhados pelo estabelecimento e/ou casas que contêm sensores de recolha e agregadores, com a responsabilidade de gerir e juntar todos os dados que estão a ser recolhidos da sub-rede pela qual estão responsáveis.

2.2.1 Equipamentos de Recolha e Controlo

Devido às várias limitações dos contadores clássicos apresentadas anteriormente, foram propostos novos equipamentos com capacidade de recolha e controlo nos seguintes módulos: electricidade, gás, higrómetro e termómetro.

Nos módulos de electricidade e gás, pretende-se analisar e guardar os gastos realizados, por forma a registar os consumos instantâneos dos consumidores. O higrómetro é usado para efectuar estudos sobre o clima, registando o nível de humidade no local onde se encontra o sensor, por forma a prevenir eventuais danos físicos. O termómetro permite recolher a temperatura no local de estudo.

Estes módulos estão conectados à WSN, associados a dispositivos simples, que têm

como funções recolher e enviar os dados para o agregador responsável.

2.2.2 Data Logger

O Data Logger (DL) é um equipamento electrónico responsável por estabelecer a ligação entre a empresa que está a prestar o serviço e a habitação ou o estabelecimento que o utiliza. Possui grande capacidade de processamento e de armazenamento de dados, comparativamente aos outros dispositivos instalados.

Este recebe todos os dados recolhidos dos dispositivos da sub-rede pela qual está responsável, servindo como agregador, trata-os e reencaminha-os para o sistema geral. Detecta ainda se todos os dispositivos estão operacionais e em caso de anomalias, deve informar o sistema geral que algo de errado está a acontecer, para que sejam efectuados todos os procedimentos técnicos necessários para a correcção da mesma.

Assim, através da inserção deste equipamento na rede, todas as manutenções e configurações serão possíveis remotamente, evitando assim a deslocação dos técnicos responsáveis ao local, nos casos em que existam correcções de anomalias ou actualizações de *software*.

2.2.3 Arquitectura do Sistema Actual

O sistema actual contém algumas falhas, que no entender da empresa fornecedora de energia, colocam em risco o bom funcionamento da tecnologia. Seguidamente, é apresentada a topologia da rede, assim como as falhas que esperam ver-se corrigidas.

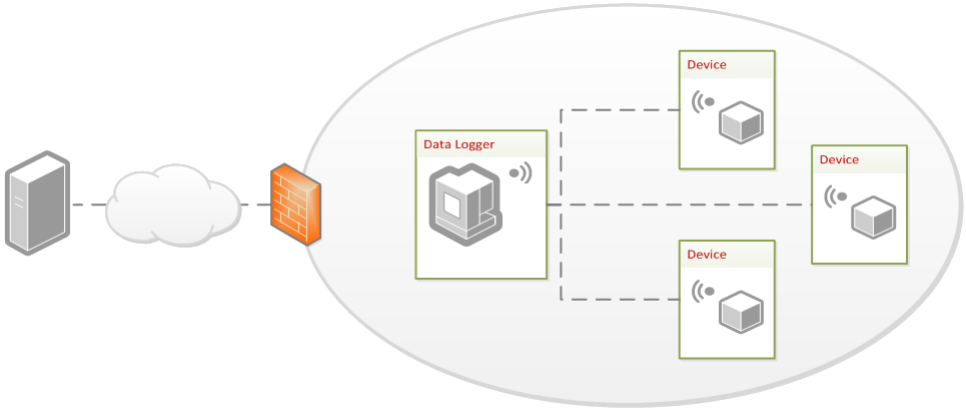

Figura 2.1: Topologia Actual

A área no qual o meu trabalho incide encontra-se na figura 2.1 destacada sob um fundo cinzento. Os dispositivos que compõem a rede de sensores são *motes* Squidbee *open-source*, desenvolvidas pela Libelium [24].

De modo genérico, esta possui três sensores de recolha incorporados (humidade, temperatura e luminosidade) e envia os dados recolhidos através de comunicações sem fios utilizando o protocolo ZigBee. A nível de *hardware*, esta *mote* consiste numa *board* Arduino e contém um XBee Shield que suporta um módulo XBee. Apesar de ser simples programá-la, vantagem inerente à plataforma Arduino, não possuí nenhum modo de economia de energia e não está pronta para fins comerciais (não certificada) [14, 27, 3].

A comunicação entre os dispositivos e o DL é feita através de portadoras rádio. Como já referido, estas comunicações estão dependentes de vários condicionalismos que influenciam directamente a transmissão ou recepção de mensagens. Neste caso em concreto, não existe nenhum mecanismo de monitorização ou controlo da rede que melhore a fiabilidade da comunicação entre os dispositivos e o DL.

O facto do conteúdo das mensagens que são enviadas ser visível, não existindo nenhum mecanismo de segurança associado e dos sensores não se identificarem perante o DL, permitindo que qualquer sensor possa estabelecer uma ligação com este e enviar dados que não sejam fidedignos, torna este sistema demasiado vulnerável para as pretensões da empresa fornecedora de energia.

É de salientar que, a informação recolhida pelos sensores, não passa por qualquer tipo filtro, de modo a verificar se os valores se encontram dentro dos padrões normais. Um exemplo prático será a temperatura ambiente de uma casa, onde a ocorrência de valores elevados (maiores que 40°C) não é normal. Assim sendo, é importante a inclusão de um mecanismo que permita este tipo de verificação.

2.3 Requisitos da Solução

Finalizada a análise à arquitectura do sistema actual e das falhas inerentes, é tempo de identificar os requisitos para a solução proposta. Estes estão divididos em duas secções: requisitos de confiabilidade e requisitos de segurança.

Os requisitos de confiabilidade referem-se a técnicas e estratégias que complementam a rede de forma a que esta tenha um bom funcionamento tanto em condições normais

como adversas, enquanto os requisitos de segurança garantem acesso exclusivo aos nós e aos dados apenas a utilizadores que possuam permissões.

2.3.1 Requisitos de Confiabilidade

2.3.1.1 Transmissão Inteligente de Mensagens

A transmissão das mensagens pode ocorrer em três momentos diferentes: a pedido do DL, periodicamente e quando o *buffer* atinge o limite.

De forma a evitar demasiadas transmissões, ocupando o canal ou consumindo energia da sua bateria (caso tenha) desnecessariamente, a transmissão de mensagens pode ser feita de forma controlada, nomeadamente, a periodicidade da transmissão pode variar consoante a aplicação.

Outro cuidado a ter em conta, é no caso em que estamos a recolher informações críticas, por exemplo, a temperatura junto de um centro de gás, em que valores demasiado altos poderão indicar um fogo próximo e consequente explosão. Nestes casos, é necessário que haja pouco atraso na transmissão e medidas contra a perda de pacotes.

2.3.1.2 Alcance entre Dispositivos

A topologia de rede varia de cenário para cenário, dependendo do número de dispositivos necessários. Por exemplo, se todos os dispositivos estiverem no alcance do coordenador basta uma rede em estrela ou simplesmente ponto-a-ponto; Nos casos em que isto não acontece, podemos ter uma topologia mais complexa como em malha ou em árvore.

Para ajudar a determinar a topologia mais adequada ao cenário, serão realizados testes em diferentes ambientes para obter a distância máxima entre os dispositivos, de forma a que a rede possa estar completamente operacional e com o menor número de falhas.

Naturalmente, se o dispositivo está localizado num espaço livre terá maior alcance do que numa indústria onde haja muito ruído electromagnético e/ou obstáculos, como por exemplo paredes.

2.3.1.3 Recolha de Dados no Sensor

A recolha de dados no sensor pode ocorrer em dois momentos diferentes: a pedido do DL e periodicamente.

A periodicidade das leituras pode ser diferente dependendo do cenário, sujeito a factores como o tipo de módulo, o estabelecimento, o cliente, entre outros. Por exemplo, leituras instantâneas da temperatura não iriam trazer dados relevantes, pois a variação da temperatura normalmente é lenta (excluindo casos em que a recolha de temperatura é um factor importante). Outro, será a mediação da corrente eléctrica numa casa ou numa empresa, obrigatoriamente terá que ser diferente.

2.3.2 Requisitos de Segurança

2.3.2.1 Confidencialidade

Todas as mensagens da rede apenas devem ser acessíveis exclusivamente por quem tem autorização, protegendo desta forma os dados recolhidos. Esta garantia está relacionada com o conceito confidencialidade.

A perda de confidencialidade pode levar terceiros a obter informações sobre os dados que estão a ser recolhidos pelos sensores de recolha e posterior aproveitamento para uso impróprio.

2.3.2.2 Integridade

Não deve ser possível a inclusão de novos dispositivos sem que eles consigam provar que merecem a confiança dos outros dispositivos.

A perda de integridade, pode levar terceiros a inserir dispositivos falsos na rede e estes terem acesso aos dados recolhidos (perda da integridade do sistema). Pode ainda acontecer que as mensagens sejam adulteradas para benefício próprio (perda da integridade das mensagens). Por exemplo, se o dono do estabelecimento conseguir adulterar o sistema, pode fazer com que o sensor de electricidade não recolha informações e desta forma não efectuar o pagamento do consumo real.

Como pode facilmente perceber-se, a perda de integridade, pode levar a prejuízos incalculáveis da empresa envolvida.

Capítulo 3

Revisão Tecnológica

Existem diversas formas de criar uma WSN, tanto a nível de *hardware*, como a nível de *software*. Normalmente um *hardware* vem equipado com *software* específico, por isso, foi necessário conhecer as duas vertentes detalhadamente para depois se poder tomar uma decisão sobre qual a tecnologia usar.

Neste capítulo, são focadas algumas tecnologias consideradas, para posteriormente ser apresentada a solução proposta.

3.1 *Hardware*

Nesta secção é apresentado todo o *hardware* considerado para a construção da solução. A escolha recai sobre o que nos parece mais adequado em termos de preço e fiabilidade. Outros aspectos importantes são o facto de ser *open-source* e ter elevada disponibilidade.

3.1.1 Data Logger

O *hardware* que de seguida é apresentado foi considerado durante a construção da solução para o DL. A especificação dos requisitos para este componente é:

- 1 porta USB, para ligação à componente rádio;
- 1 porta Ethernet, para permitir acesso ao exterior;

- Unidade de processamento, memória e armazenamento;
- Sistema operativo e *software* compatível com a componente de rádio.

Seria ainda interessante que este *hardware* contemplasse:

- Dimensões reduzidas;
- Invólucros robustos;
- Baixo consumo.

Tendo em conta os requisitos definidos, são apresentados, respectivamente, o Raspberry Pi, o APC e o DreamPlug. Como apoio, na tabela 3.1 temos a comparação entre os considerados, onde apenas são focadas as características que nos interessam, pois estes possuem mais do que as estritamente necessárias. Existiam outras alternativas, no entanto, estas pareceram ser as mais adequadas ao projecto.

3.1.1.1 Raspberry Pi

O Raspberry Pi é um mini PC disponível em dois modelos (A e B), barato (cerca de 20 e 29 € respectivamente) e criado pela The Raspberry Pi Foundation [12].

Este projecto vem equipado com um processador ARM1176JZF-S de 700 MHz, com 256 Megabyte (MB) de memória Random-Access Memory (RAM) e *slots* para cartões de armazenamento de dados (SD, MMC e SDIO), pois este não inclui memória não-volátil, como um disco rígido. A nível de *software*, existem versões para Debian GNU/Linux, Fedora, Arch Linux [12].

As principais diferenças entre o modelo A e o modelo B são a existência de mais uma porta USB e a presença de uma porta Ethernet no modelo B (ver tabela 3.1).

As principais vantagens desta placa relativamente ao APC e ao DreamPlug, apresentados nas sub-secções seguintes, são o preço e o tamanho. Apesar das capacidades de processamento e de armazenamento serem inferiores a todas as alternativas apresentadas, são mais do que suficientes para a aplicação (ver tabela 3.1).

Sendo esta alternativa bastante apelativa, apenas o modelo B nos interessaria, porque um dos requisitos necessários é a porta Ethernet. Possui as desvantagens de necessitar de uma caixa para a proteger e de um cartão para armazenamento, pelo que iria

aumentar consideravelmente o preço. Acrescido a estes problemas, a falta de *stock* para a compra levou a que esta possibilidade não fosse usada.

3.1.1.2 APC

Depois do aparecimento do projecto Raspberry Pi, a VIA Technologies disponibilizou recentemente o seu mini PC por cerca de 40 € denominado por APC [29].

Este projecto vem equipado com um processador Chip VIA de 800MHz, com 512 MB (DDR3) de memória RAM e 2 GB NAND de memória *flash* para armazenamento de dados. Relativamente ao *software*, está equipado com Android 2.3 [29].

A principal vantagem desta placa relativamente ao *hardware* apresentado é o consumo. Apesar de ser um pouco mais cara que o Raspberry Pi (no entanto é computacionalmente melhor), não é uma possibilidade a descartar (ver tabela 3.1).

Esta placa, tal como o Raspberry Pi, necessita de uma caixa no formato Neo-ITX, compatível com os *chassis* Mini-ITX e micro ATX. Como é um projecto recente, a sua aquisição também não foi possível.

3.1.1.3 DreamPlug

O DreamPlug é um mini PC criado pela GLOBALSCALE Technologies, Inc. Este vem equipado com um processador Marvell Sheeva Core Embedded CPU de 1.2 GHz, com 512 MB (DDR2 800 MHz) de memória RAM e um cartão SD interno com 4 GB para o *kernel* e sistema de ficheiros. A nível de *software*, existem versões para Debian e Ubuntu [9, 13].

Comparativamente ao *hardware* apresentado, tem como principais vantagens ser um modelo final, não tendo custos adicionais (p.e. compra de uma caixa ou cartão para armazenamento de dados) e possuir maior capacidade de processamento e de armazenamento (ver tabela 3.1). Por este motivo, é também a solução mais cara, cerca de 155 €, já com Joint Test Action Group (JTAG) incluído [9].

Este é o modelo escolhido para a construção do protótipo, porque dentro dos considerados, foi o único que foi possível adquirir no devido prazo. No entanto, as alternativas apresentadas devem ser tomadas em conta, caso se pretenda substituir o equipamento.

Tabela 3.1: Comparação entre o *hardware* considerado para o DL

			Raspberry Pi	
Características	DreamPlug	APC	Modelo A	Modelo B
Sistema Operativo	Ubuntu/Debian	Android 2.3	Debian GNU/Linux, Fedora, Arch Linux	
Memória	512 MB 16-bit DDR2	512 MB @ DDR 3	256 MB (GPU partilhado)	
Armazenamento	4 GB micro-SD	2 GB NAND Flash	*Slot* para cartões SD, MMC, SDIO	
CPU	Marvell Sheeva Core Embedded CPU @ 1.2 GHz	VIA 800 MHz	700 MHz núcleo ARM 1176JZFS (ARM 11)	
Portas USB 2.0	2	4	1	2
Rede LAN	2 x Gigabit Ethernet 10/100/1000 Mbps	Ethernet 10/100	Não disponível	Ethernet 10/100 (RJ45)
Alimentação	-	-	5 *volts* via Micro-USB ou GPIO *header*	
Dimensões (mm) (WxH)	108 x 58	170 x 85 (Neo-ITX *standard)*	85.60 x 53.95	
Preço	155 €	40 €	20 €	29 €

Referências: [29, 12, 13]

3.1.2 Dispositivos de Rede

O *hardware* que de seguida é apresentado foi considerado durante a construção da solução para os dispositivos de rede. A especificação dos requisitos para este componente é:

- Bom alcance;
- Boa taxa de transferência de dados;
- Baixo consumo.

Tendo em conta os requisitos definidos e não sendo possível avaliar todos os dispositivos de rede concorrentes, devido à enorme quantidade de dispositivos disponíveis no mercado. Por este motivo, o preço teve um papel preponderante na escolha, assim como o protocolo de comunicação utilizado. Nas sub-secções seguintes, são apresentados, respectivamente, o modelo IRIS da Crossbow e a junção de uma placa Arduino com o módulo de rádio XBee. Como apoio, na tabela 3.2 temos a comparação entre os mesmos.

3.1.2.1 Crossbow Motes

A Crossbow Technologies oferece um amplo portefólio de produtos para redes de sensores, como os modelos MICAz, TelosB, IRIS e ainda, *kits* de desenvolvimento. Estas *motes* executam o TinyOS [2] - sistema operativo aberto projectado para baixo consumo de energia em dispositivos sem fios - que tem como principais funções gerir a potência de transmissão da componente RF e a transparência da rede para o utilizador. Neste caso em concreto, apenas é destacado o modelo IRIS, pois este é o último módulo da Crossbow Technologies e incluí várias melhorias em relação às outras, nomeadamente o aumento do alcance de transmissão [15].

A nível da componente rádio, este modelo tem um alcance interior de 50 metros, um alcance exterior de 300 metros e uma taxa de transferência de dados de 250 Kbps. Apesar das características da IRIS serem relativamente melhores comparativamente ao *hardware* apresentado de seguida (ver tabela 3.2), nomeadamente o alcance, também apresenta algumas desvantagens como a impossibilidade de alterar a componente rádio (caso seja necessário) e o seu custo ser mais elevado. Por esta razão, o *hardware* escolhido foi o Arduino e XBee em conjunto, explicado na secção seguinte.

3.1.2.2 Arduino e XBee

O Arduino é uma placa de controle I/O baseada no micro-controlador Atmega, que tem como objectivo servir de controle para outros sistemas [27].

Com esta placa, é possível construir as mais diversas aplicações, desde muito simples até às mais complexas. Em [21], podemos consultar algumas das inúmeras aplicações onde é usada. A facilidade com que é programada e com que são adicionados sensores, LEDs, mostradores, relés, motores e outros dispositivos nos pinos I/O são os principais factores do seu sucesso. Em [28], temos todos os modelos deste projecto.

Os módulos XBee, fabricados pela Digi International, incluem todo o *hardware* e a lógica necessária para implementar uma rede ZigBee - Norma de protocolos de comunicação para redes sem fios baseados em baixa taxa de transmissão, curto alcance, baixo consumo de energia e custo de implementação [16, 6].

São fabricados em duas versões, XBee e XBee-Pro, que se diferenciam sobretudo na potência de emissão e sensibilidade de recepção, logo no alcance máximo. Em [6], podemos consultar as principais características das duas versões, assim como uma comparação entre elas.

A utilização destes dois produtos em simultâneo é possível através dos modelos Arduino Uno ou Arduino Fio. Neste caso em concreto, apenas é usado o Arduino Fio, pois a sua aquisição é mais vantajosa monetariamente. Este modelo já integra uma *socket* que possibilita a comunicação com o módulo XBee. Relativamente à componente rádio, a versão escolhida é a XBee. Caso se pretenda aumentar as capacidades rádio, basta apenas trocar o módulo para a versão XBee-Pro [6, 28].

A escolha relativa aos dispositivos de rede recai sobre este *hardware*, não só por ser a solução mais económica, mas também por maior suporte e facilidade de programação.

3.2 Comunicações

Escolhido o *hardware*, interessa perceber agora todos os pormenores relativos às características a nível de comunicações. À medida que são mostradas algumas características da norma ZigBee, ficaremos a perceber a sua importância para solucionar alguns dos problemas que é necessário resolver.

Tabela 3.2: Comparação entre o *hardware* considerado, a nível da componente rádio, para os dispositivos de rede

Características	IRIS Mote	XBee
Banda de Frequência (GHz)	2.4	2.4
Alcance Interior (m)	50	40
Alcance Exterior (m)	300	120
Transferência de Dados (Kbps)	250	250
Potência de Transmissão (dBm)	3 (typ)	$+3^{*}$, $+1^{**}$
Sensibilidade do Receptor (dBm)	-101 (typ)	-96^{*}, -95^{**}

Referências: [15, 6]

* Modo de impulso ligado

** Modo de impulso desligado

3.2.1 Norma ZigBee

O ZigBee é uma norma *standard* para comunicações sem fios de baixa transmissão de dados, custo, consumo e complexidade, desenvolvido pela ZigBee Alliance. Este opera em 3 frequências distintas [8]:

- 2.4 GHz com débito de 250 KB/s (uso global, modulação Offset Quadrature Phase-Shift Keying (O-QPSK));
- 915 MHz com débito de 40 KB/s (USA e Austrália, modulação Binary Phase-Shift Keying (BPSK));
- 868 MHz com débito de 20 KB/s (Europa, modulação BPSK).

O ZigBee foi criado especificamente para redes de sensores, permitindo o uso de grandes quantidades de dispositivos (máximo de 65535 dispositivos por cada coordenador). Devido às suas características, destina-se a aplicações que necessitam do uso de bateria de longa duração. A taxa transferência de dados máxima é 250 *Kbits* por segundo [8].

Fazendo uma breve comparação entre o ZigBee e outras tecnologias de redes sem fios, nomeadamente o Bluetooth e a norma Institute of Electrical and Electronics Engineers (IEEE) 802.11, é de salientar que estes são mais complexos de utilização mais comum, e que por operarem na mesma banda (2.4 GHz), interferem directamente na qualidade de serviço do ZigBee (ver secção 4.1).

Na tabela 3.3 é feita uma comparação entre as tecnologias supracitadas relativamente à taxa de transferência de dados, ao alcance interior e a algumas aplicações. Posteriormente, na figura 3.1 é apresentada a relação entre a energia consumida, a complexidade e custo relativamente à taxa de transferência de dados [8].

Tabela 3.3: Transmissão de dados, alcance e aplicações das tecnologias ZigBee, Bluetooth e IEEE 802.11

Protocolo	Transmissão de Dados	Alcance (m)	Possíveis Aplicações
IEEE 802.11	Elevada	50 - 100	Internet sem fios
Bluetooth	Baixa	2 - 10	Ratos e Teclados sem fios
ZigBee	Muito Reduzidas	10 - 100	Redes de Sensores

Referências: [16]

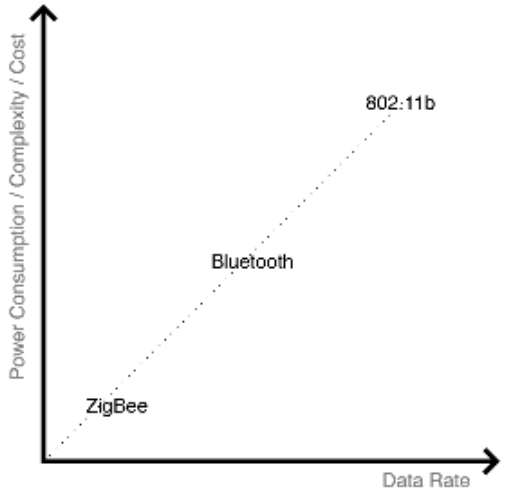

Figura 3.1: Consumo, complexidade e custo das tecnologias ZigBee, Bluetooth e IEEE 802.11 (Adaptado de [16])

Como podemos facilmente verificar, as tecnologias não são concorrentes, isto é, as suas aplicações são bem diferenciadas. O Bluetooth é mais indicado para aplicações que requerem um maior débito, como por exemplo ratos e teclados sem fios. A norma IEEE 802.11, onde temos uma elevada transmissão de dados comparativamente aos anteriores e onde o seu custo, complexidade e consumo são elevados, pode ser encontrado nas redes sem fios de nossas casas.

3.2.2 Tipos de Dispositivos

Existem dois tipos de dispositivos numa rede sem fios IEEE 802.15.4: Full-Function Device (FFD) e Reduced-Function Device (RFD). Um dispositivo FFD possuí todas as capacidades descritas na norma IEEE 802.15.4, com a capacidade de aceitar qualquer regra e de comunicar com qualquer dispositivo da rede [8].

Por outro lado, um RFD é um dispositivo limitado, que comunica somente com um único FFD e é usado apenas em aplicações muito simples, pois tem capacidades de processamento e de armazenamento reduzidas, normalmente muito inferiores às dos dispositivos FFD [8].

3.2.3 Pilha Protocolar

O ZigBee possui uma pilha protocolar bastante simplificada. Este não se encaixa exactamente no modelo OSI (7 camadas), apesar de conter algumas camadas iguais, nomeadamente as camadas Physical Layer (PHY), Medium Access Control (MAC) e a Network Layer (NWK). As restantes camadas do modelo OSI (transporte, sessão, apresentação e aplicação) são envolvidas nas camadas Application Support Sublayer (APS) e ZigBee Device Object (ZDO) [11].

Na pilha protocolar (ver figura 3.2), a camada mais baixa é a PHY. Esta é a camada mais próxima do *hardware* e serve para controlar e comunicar com o *transceiver radio*. É responsável por activar o *transceiver radio* que transmite ou recebe pacotes. Selecciona ainda a frequência do canal que vai ser usado e assegura que este não é usado por outros dispositivos noutra rede [8].

A camada MAC é responsável por gerar *beacons* e sincronizar o dispositivo para os *beacons* (numa rede de *beacons* activos). Fornece ainda os serviços de associação e de dissociação [8].

O ZigBee adoptou da norma IEEE 802.15.4 a camada PHY e a MAC, como é visível na figura 3.2, por isso, um dispositivo compatível com ZigBee é também compatível com a norma IEEE 802.15.4 [8].

A camada NWK é responsável pela gestão e encaminhamento da rede. Encaminhamento é o processo de selecção da rota que uma mensagem vai seguir até chegar ao dispositivo de destino. O coordenador da rede e os *routers* são responsáveis por descobrir e manter as rotas da rede. Um dispositivo terminal não pode realizar este

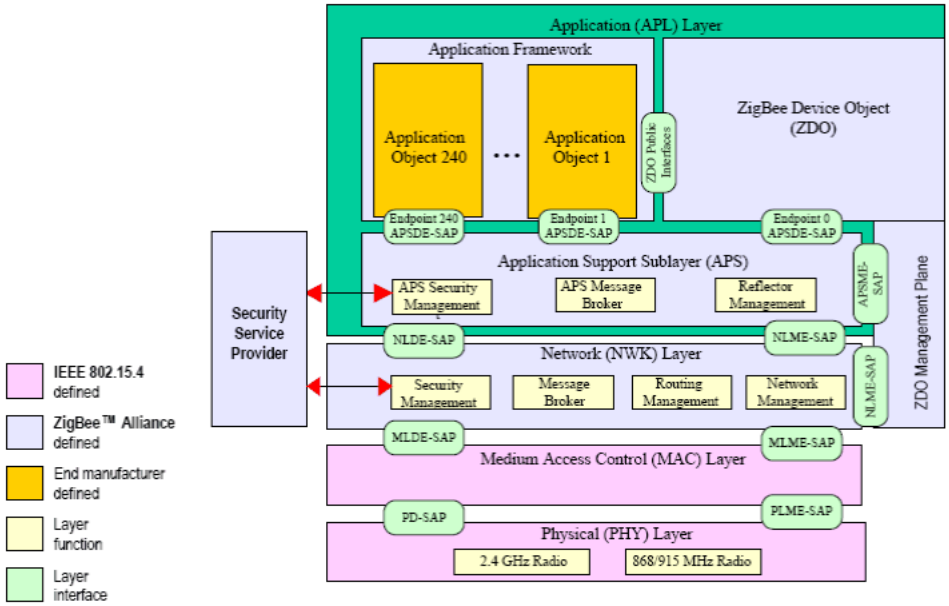

Figura 3.2: Pilha protocolar (de [4])

processo. A camada NWK do coordenador da rede é responsável por criar uma nova rede e seleccionar a topologia de rede. Este atribui os endereços NWK para todos os dispositivos da rede [8].

A camada Application Layer (APL) contém a sub-camada APS, o ZDO e a Application Framework (AF). É a camada mais alta do protocolo e alberga os objectos da aplicação. Os fabricantes desenvolvem os objectos da aplicação para personalizar um dispositivo para várias aplicações. Um objecto permite controlar e gerir as camadas do protocolo [8].

A sub-camada APS fornece um *interface* entre a camada NWK e a camada APL. Funciona como um filtro para as aplicações a correr acima desta, de forma a simplificar a lógica dessas aplicações. Filtra ainda mensagens duplicadas que foram enviadas pela camada NWK. A camada APS mantém uma tabela *binding* local, que indica os dispositivos de rede com quem quer falar [11].

O ZDO é responsável pela gestão local e remota da rede. Fornece serviços para descobrir outros dispositivos e serviços na rede, e é responsável pelo estado actual do dispositivo na rede [11].

Por fim, a AF contém a *ZigBee Cluster Library* e fornece um *framework* onde as aplicações são executadas [11].

3.2.4 Regras dos Dispositivos

Numa rede IEEE 802.15.4, um dispositivo FFD pode ter 3 diferentes regras: Personal Area Network (PAN) *Coordinator*, *Coordinator* e *Device*. Um *Coordinator* é capaz de retransmitir mensagens para outros dispositivos da rede. Quando o *Coordinator* tem capacidade para controlar a PAN designa-se por PAN *Coordinator* e é único nessa rede. Um dispositivo que não se comporte como *Coordinator*, denomina-se por *Device* [16].

Tabela 3.4: Comparação entre os tipos de dispositivos ZigBee e IEEE 802.15.4

Protocolo	Tipo de Dispositivo		
IEEE 802.15.4	PAN *Coordinator*	*Coordinator*	*Device*
ZigBee	ZigBee Coordinator (ZC)	ZigBee Router (ZR)	ZigBee End Device (ZED)

Referências: [16]

Comparativamente à norma IEEE 802.15.4, como é visível na tabela 3.4, o ZigBee usa uma terminologia um pouco diferente. Uma rede ZigBee define três tipos de dispositivos: ZC, ZR e ZED [16].

Um ZC possui as seguintes propriedades: é capaz de seleccionar um canal e o PAN ID para iniciar a rede; permite que ZR e ZED se juntem à rede; pode auxiliar no encaminhamento de dados; não possuí modo de poupança de energia, pelo que deve estar sempre alimentado; e pode ainda, armazenar mensagens Radio-Frequency (RF), permitindo que os ZED entrem em modo poupança de energia.

No que diz respeito ao ZR, este possui características semelhantes a um ZC. A principal distinção entre ambos é o facto de necessitar de se juntar a uma ZigBee PAN antes de transmitir, receber ou encaminhar dados.

Um ZED é o dispositivo mais simples da rede e por isso, diferencia-se mais das propriedades dos anteriores. Este necessita de se juntar a uma ZigBee PAN antes de transmitir ou receber dados tal como o ZR, no entanto, não permite que dispositivos se juntem à rede. Para transmitir e receber dados requer sempre o seu antecessor (ZC ou ZR, dependendo da topologia de rede), sendo que não faz encaminhamento de dados e ainda, pode entrar em modo de poupança de energia e ser alimentado por baterias.

3.2.5 Topologias de Rede

A topologia de rede descreve o *layout* da mesma, demonstrando como os dispositivos se encontram interligados. Pode ser descrita fisicamente, que consiste na verdadeira disposição física, ou logicamente, descrevendo o fluxo de dados. Existem várias formas de organizar os dispositivos uns com os outros. De seguida, são apresentados os esquemas mais vulgares numa rede de sensores ZigBee (ver figura 3.3) [7]:

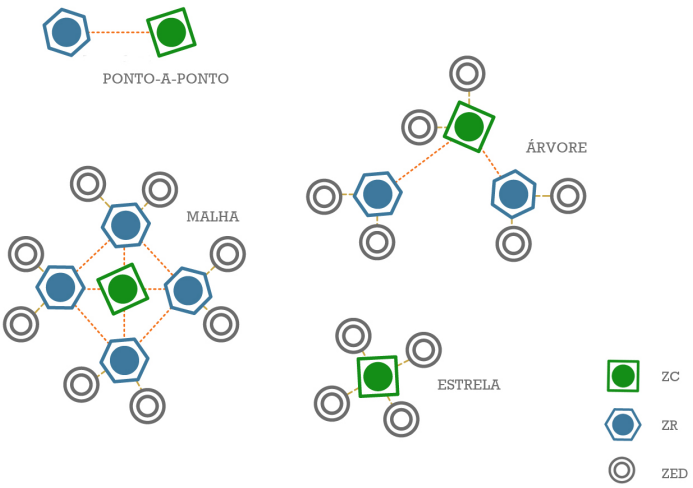

Figura 3.3: Topologias de Rede (Adaptado de [7])

- ***Ponto-a-ponto:*** Rede simples com apenas dois dispositivos rádio. Um deles necessita de ser ZC para que a rede seja formada; o outro pode ser configurado como ZR ou ZED.
- ***Estrela:*** O ZC encontra-se centrado na topologia e conecta-se a um circulo de ZED. Todas as mensagens na rede passam pelo ZC, que faz o encaminhamento necessário entre os dispositivos. Os ZED não comunicam entre si directamente.
- ***Malha:*** Este tipo de configuração é complexa e requer dispositivos ZR ligados ao ZC. Os ZR fazem o encaminhamento entre o ZC e os ZED e podem ser responsáveis por vários ZED. Os ZED geram e recebem a informação, mas necessitam dos nós adjacentes para comunicarem com os outros dispositivos na rede.
- ***Árvore:*** É uma derivação da topologia em malha, onde os ZED se agrupam à volta dos ZR.

Uma topologia adequada pode resolver muitos problemas, no entanto a determinação desta não é fácil. A disposição dos dispositivos pode variar muito, estando directamente relacionado com a distância total de cobertura e as condições no local. A cobertura de 200 m^2 num centro comercial necessita de mais dispositivos que a mesma distância num parque com vegetação (p.e. muito do sinal é perdido a atravessar obstáculos).

Neste sentido, a utilização destas topologias e de boas práticas relacionadas com a construção da topologia vai contribuir e muito para a fiabilidade da rede. Para se perceber melhor esta dificuldade, na secção 4.1 temos um estudo que foi efectuado durante a realização deste projecto.

3.2.6 Endereços Simples, Endereços PAN e Canais de Comunicação

Numa rede ZigBee encontramos três tipos de endereços que são responsáveis por localizar o dispositivo. Na tabela 3.5, temos um exemplo dos endereços que encontramos num módulo.

Tabela 3.5: Endereços ZigBee

Tipo de Endereço	Exemplo	Endereço Único
64 *bits*	0013A2004071CE0B	Sim, sempre e em qualquer lugar
16 *bits*	F9AB	Sim, mas apenas na própria rede
Identificador	COORDENADOR	Não é garantido

Referências: [7]

O primeiro endereço é único e identifica o dispositivo. Por sua vez, o endereço de 16 *bits* é atribuído dinamicamente pelo coordenador da rede (ZC) e é único apenas nessa rede; Poderá existir outra rede com o mesmo identificador. Finalmente, o identificador do dispositivo é atribuído pelo técnico que configura os dispositivos. Este identificador é usado apenas para que os humanos o possam identificar e distinguir facilmente. Recorrendo a nova analogia, acontece o mesmo com um endereço IP, em que o atribuímos a um nome (Domain Name System (DNS)) para o identificar [7].

Um endereço de 16 *bits* define exclusivamente uma PAN. Apenas módulos RF com o mesmo PAN ID podem comunicar entre si. Na figura 3.4 podemos visualizar a organização dos endereços numa rede ZigBee [7].

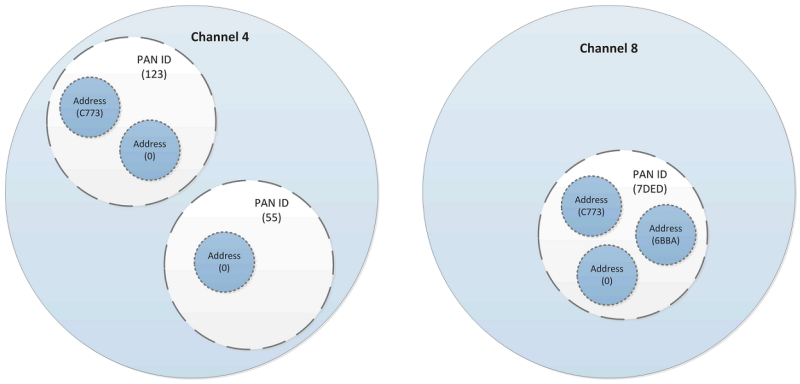

Figura 3.4: Organização dos endereços (Adaptado de [7])

Para que a comunicação seja perfeita, os módulos rádio têm que estar sincronizados na mesma frequência. Quando o ZC procura pelo endereço PAN, verifica todos os canais disponíveis e utiliza um para a comunicação na rede. Todos os módulos rádio da rede devem usar o mesmo canal [7].

3.2.7 Encaminhamento

O algoritmo de encaminhamento usado pelo ZigBee é baseado no modelo Distance Vector (DV), onde um ZR é responsável por dispersar pacotes da origem para um determinado destino, guardando o próximo salto na sua tabela de encaminhamento. A entrada que é adicionada contém o endereço do próximo salto e representa a distância lógica para o destino [20].

A rota para o destino é determinada usando o processo *route discovery*, onde o dispositivo de origem envia um pedido *route request* para o endereço *broadcast* e o dispositivo de destino, quando recebe esse pedido, envia de volta um *route reply* [20].

Sempre que chega um pacote a um dispositivo, este verifica na sua tabela de encaminhamento se existe alguma entrada para o destino pretendido. Caso exista, este reencaminha o pacote para o próximo salto [20].

Este mecanismo permite que facilmente a rede se adapte às várias circunstâncias. Na eventualidade de um nó falhar, a rede reorganiza-se para tentar colmatar a falha. A inclusão de um novo nó é feita de forma automática, com a reorganização das tabelas de encaminhamento, evitando novas configurações. Isto é uma mais valia para os objectivos do projecto.

3.2.8 Segurança

A segurança neste tipo de redes é essencial quando as comunicações ou a rede precisam de ser protegidas. A sua inclusão deve ser estritamente utilizada apenas em aplicações que realmente precisam, porque é uma operação pesada e que requer muitos recursos. Exemplo da sua utilização serão aplicações que envolvam dados confidenciais [7].

O ZigBee possui vários níveis de segurança, que podem ser configurados de acordo com as necessidades das aplicações. Contém [6]:

- Duas chaves de segurança que podem ser pré-configuradas ou obtidas durante a inserção do novo dispositivo;
- Centro de segurança (ver secção 3.2.8.2);
- Mecanismos que garantem integridade, confidencialidade e autenticação de mensagens.

A segurança é aplicada nas camadas NWK e APS (ver figura 3.2). Os pacotes são cifrados com o algoritmo Advanced Encryption Standard (AES) (algoritmo de chave simétrica, ou seja, é utilizada a mesma chave para cifrar e decifrar os dados) e chaves de tamanho 128 *bits*. As chaves de rede e de ligação são opcionais e usadas para cifrar os dados. Apenas dispositivos com as mesmas chaves estão habilitados a comunicar na rede [6].

De forma a garantir os requisitos de segurança apresentados na secção 2.3.2, a inclusão dos mecanismos de segurança que de seguida são apresentados são necessários na solução final.

3.2.8.1 Chaves de Segurança

Numa rede ZigBee, existem 3 tipos chaves de segurança: mestra, rede e ligação. As chaves de rede e de ligação podem ser usadas simultaneamente:

- ***Chave mestra:*** Não é usada para cifrar conteúdos. Em vez disso, serve como sendo um segredo que é partilhado entre 2 dispositivos durante o estabelecimento de chaves Symmetric-Key Key Exchange (SKKE), para gerar as chaves de ligação [19].

- ***Chave de rede:*** Usada para cifrar a camada APS e os dados da aplicação (ver figura 3.5). Esta chave permite segurança salto a salto. Cada pacote é cifrado, autenticado e enviado para o próximo salto. Chegado a esse nó, este decifra-o, autentica-o e no caso deste pacote não se destinar a esse nó, é novamente cifrado, autenticado e enviado para o próximo salto. Este mecanismo repete-se até que o pacote chegue ao nó de destino [7].

  Isto adiciona latência ou atraso na transmissão de pacotes, assim como a adição de 18 *bytes*. Por consequência, o tamanho máximo do *payload* diminui. Todos os dispositivos na rede partilham esta chave [7].

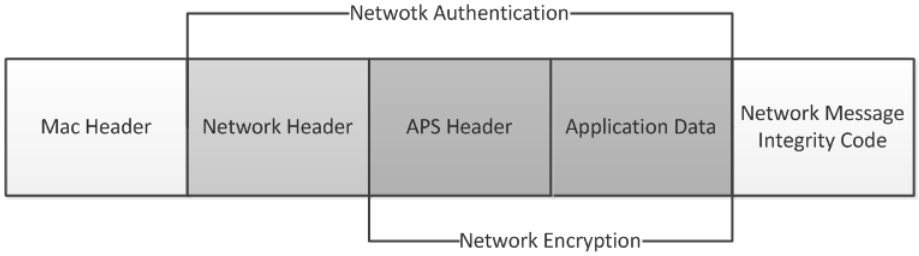

Figura 3.5: Camada de rede (de [6])

O cabeçalho NWK, o cabeçalho APS e os dados da aplicação são todos autenticados com o algoritmo AES e chave de tamanho 128 *bits* (ver figura 3.5). É gerado um *hash* com estes campos e adicionado ao fim do pacote 4 *bits* com o Message Integrity Code (MIC) gerado. O MIC permite verificar que as mensagens não foram modificadas, garantindo a sua integridade. Se o dispositivo recebe um pacote e o MIC não coincide com o MIC calculado, este pacote é descartado [6].

- ***Chave de ligação:*** Usada para garantir segurança nas mensagens *unicast* entre dois dispositivos na camada de aplicação (ver figura 3.6). Garante protecção extremo-a-extremo, isto é, o pacote é cifrado na origem e decifrado apenas no destino [7].

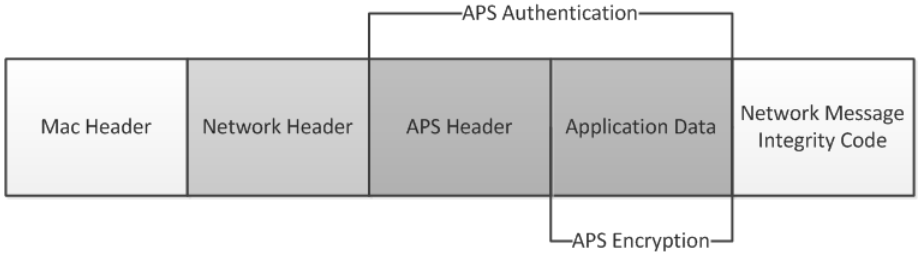

Figura 3.6: Camada APS (de [6])

O cabeçalho APS e os dados da aplicação são todos autenticados com o algoritmo AES e chave de tamanho 128 *bits* (ver figura 3.6). É gerado um *hash* com estes campos e adicionado ao fim do pacote 4 *bits* com o MIC gerado. O MIC permite verificar que as mensagens não foram modificadas, garantindo integridade de mensagens. Se o dispositivo recebe um pacote e o MIC não coincide com o MIC calculado, este pacote é descartado [6].

3.2.8.2 Centro de Confiança

O centro de confiança autentica a entrada de novos dispositivos na rede. Este é responsável por actualizar e enviar as chaves de rede.

Normalmente, o centro de confiança é o coordenador da rede (ZC), mas esta tarefa pode ser atribuída a um outro dispositivo. É responsável pelas seguintes regras de segurança [19]:

- **Gerir confiança**, para autenticar os dispositivos que requerem entrada na rede;
- **Gerir rede**, para manter e distribuir as chaves de rede;
- **Gerir configurações**, para permitir segurança extremo-a-extremo entre dispositivos.

3.2.8.3 Frame Counter

O cabeçalho de pacotes de rede cifrados inclui um *frame counter* de 32 *bits*. Cada dispositivo na rede mantém um contador de 32 *bits* que é incrementado a cada transmissão. Se um dispositivo recebe um pacote de um vizinho com o *frame counter* menor do que o anterior recebido, este pacote é descartado. Este contador é usado para evitar ataques [6].

Se o contador atingir o limite, este não volta ao valor zero e terminam as transmissões. Devido ao tamanho do contador, é pouco provável que este venha a atingir o valor máximo na maioria das aplicações [6].

Para limpar o contador sem comprometer a segurança da rede, a chave de rede deve ser alterada. Quando a chave de rede é actualizada, os contadores de todos os dispositivos na rede são limpos, voltam ao valor zero [6].

3.2.9 Retransmissões

Qualquer rede sem fios tem um problema comum: a fiabilidade da rede. Tipicamente para colmatar este problema, são utilizadas transmissões multi-salto por forma a evitar demasiada perda de dados. A transmissão multi-salto obtém-se com a inclusão de novos nós em que a principal função destes é retransmitir os pacotes recebidos. Neste sentido, o ZigBee inclui pacotes Acknowledgment (ACK) para as camadas MAC e APS.

Quando um pacote é transmitido para um dispositivo remoto, este pode passar por múltiplos dispositivos. Um pacote é transmitido para o seu vizinho e um ACK é enviado na direcção oposta indicando que a transmissão desse pacote foi feita com sucesso. Caso este ACK não seja recebido, o dispositivo de origem retransmite o mesmo pacote até quatro vezes. Este ACK é designado por Mac Layer Acknowledgment [6].

O dispositivo de origem aguarda sempre por um ACK vindo do dispositivo de destino. Esse ACK passa pelos mesmos dispositivos na rede, mas com a direcção oposta. Se o dispositivo de origem não recebe esse ACK, este retransmite o pacote até duas vezes. Este ACK é designado por ZigBee APS Layer Acknowledgment [6].

3.3 Fiabilidade da Rede

Nesta secção são identificados alguns factores que influenciam directa ou indirectamente a fiabilidade da rede. Para cada factor, são descritas possíveis formas de mitigação.

Como a topologia de rede pode variar de meio para meio, sendo que a solução final não será apenas para um caso específico, estes métodos tornam-se fulcrais para a construção de uma solução fiável.

3.3.1 Alcance dos Dispositivos

Naturalmente, cada dispositivo sem fios tem sempre associado um alcance máximo de cobertura. Este varia de marca para marca e com as condições em que opera (p.e. ruído electromagnético, interferências) e mesmo na própria marca podemos encontrar modelos com alcances diferentes (como podemos verificar na secção 3.1.2.2). Portanto, o alcance do dispositivo deve ser conhecido pelo técnico que está a fazer a instalação.

O alcance de um dispositivo está dependente de diversos factores, que por vezes tornam complicado o seu cálculo. O desempenho do módulo rádio, a sua constituição física, o formato da antena e o meio onde se encontra são factores que influenciam este cálculo [18].

Mitigação: Se as distâncias esperadas entre os dispositivos da aplicação são demasiado grandes para proporcionar diversidade nas ligações, as melhores soluções são [18]:

1. Incluir novos dispositivos na rede;
2. Aumentar o intervalo de confiança;
    - Usando antenas diferentes;
    - Amplificando o sinal, quer na transmissão, para aumentar a potência de saída, quer na recepção para aumentar a sensibilidade.

3.3.2 Densidade da Rede

Outro factor que afecta a fiabilidade da rede é a densidade. Entende-se por densidade de rede o número de dispositivos por unidade de área, onde a unidade usada é o quadrado do seu alcance [18].

Na figura 3.7, temos uma rede onde a sua densidade é razoável. Podemos visualizar que o alcance dos dispositivos não cobre a rede na totalidade. Uma rede com este formato, permite várias comunicações em simultâneo.

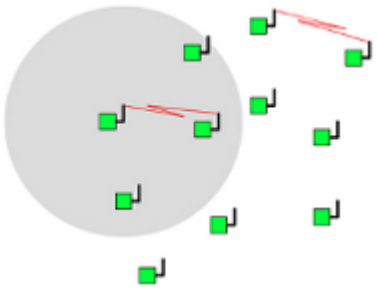

Figura 3.7: Boa densidade de rede (de [18])

Na figura 3.8, temos uma rede em que o alcance dos dispositivos cobre na totalidade a rede. Assim, apenas um par de dispositivos pode comunicar ao mesmo tempo.

Mitigação: Por vezes assumimos que a melhor solução nestes casos é aumentar a potência para a transmissão na componente rádio. Se por vezes esta é a única solução,

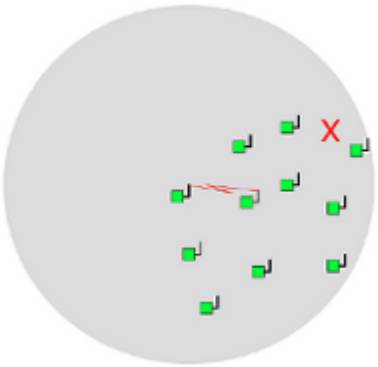

Figura 3.8: Excessiva densidade de rede (de [18])

esta não é garantia de fiabilidade e robustez da rede. No caso em que temos redes densas, diminuir a potência pode ser uma boa política, de forma a reduzir o número de dispositivos cobertos [18].

3.3.3 Utilização da Rede

O uso inadequado da rede pode levar a inúmeros problemas, tais como atraso, baixa taxa de transferência ou congestionamento do canal que, por consequência, leva ao congestionamento da rede.

A utilização da rede deve ser feita de forma moderada evitando troca de mensagens desnecessárias. Assim, uma boa utilização da rede pode contribuir para o sucesso na transmissão de dados, usufruindo melhor das totais capacidades da rede.

Mitigação: Reduzir o número de mensagens trocadas apenas ao estritamente necessário para o funcionamento da aplicação.

3.3.4 Materiais

A densidade dos materiais influencia directamente a transmissão de dados nas comunicações sem fios. Naturalmente, um corpo com uma densidade elevada vai demorar mais tempo a ser atravessado, podendo mesmo fazer com que a onda se perca. Temos ainda matérias que possuem propriedades como a reflexão, atenuação e espalhamento de sinal que modificam o sinal original.

Mitigação: Ter especial atenção na colocação dos dispositivos de rede. As boas práticas

são[1]:

1. Não colocar os dispositivos rádio próximos de matérias com estas propriedades;
2. Evitar colocar os dispositivos rádio, tanto quanto possível, em locais onde predominam estes materiais.

[1]Estas propriedades podem alterar significativamente o alcance da componente rádio do dispositivo, pelo que as formas de mitigação usadas na secção 3.3.1 podem também ser aplicadas aqui.

Capítulo 4

Testes ao Módulo de Comunicações

Neste capítulo são descritos alguns testes ao módulo de comunicações, nomeadamente à componente rádio e à rede, realizados o durante decorrer deste projecto. O objectivo é ter consciência de qual o comportamento da rede em condições mais adversas e perceber se os resultados obtidos são os esperados.

Posto isto, inicialmente são testados os módulos XBee para determinar, entre outros, o seu alcance e a sua taxa de transmissão em diferentes ambientes. É verificada ainda a existência de algumas características nestes módulos, nomeadamente quais as topologias de rede permitidas, a reacção da rede a falhas e por fim, que tipo de segurança trazem pré-configurada.

4.1 Módulos XBee

As propriedades chave da componente rádio nas aplicações práticas são o alcance e a largura de banda. O alcance refere-se à área de cobertura e a largura de banda à capacidade e rapidez de dados transferidos.

4.1.1 Parâmetros Testados

As propriedades avaliadas neste teste são:

- Taxa de recepção de pacotes;
- Intensidade de sinal (Received Signal Strength Indicator (RSSI));

- Taxa de transmissão de pacotes;
- Tempo de resposta (Round Trip Time (RTT)).

A **taxa de recepção de pacotes** permite obter uma medida de confiança sobre a transmissão de dados, que está relacionada com vários factores, como intensidade de sinal, interferências a que está sujeito ou o local onde se encontra.

A **qualidade de sinal** é o valor da intensidade de sinal durante a recepção dos pacotes. Este valor permite uma boa medida para a área de cobertura e poderá usar-se para configurar o ganho da potência de transmissão. No entanto, esta medida não nos garante que a taxa de pacotes recebidos com sucesso seja a mais elevada, pois pode acontecer que a qualidade de sinal seja boa e os pacotes estejam frequentemente a perder-se por causa, por exemplo, do canal se encontrar congestionado. Apesar disso, a qualidade de sinal é muito útil para avaliar o alcance do dispositivo.

A **taxa de transmissão** e o **tempo de resposta** são importantes para medir a quantidade de dados transmitidos e determinar o tempo gasto na transmissão de dados da aplicação. Estão dependentes da taxa de recepção de pacotes.

Os testes que são descritos de seguida são de extrema relevância para o sistema, pois contribuem para uma melhor compreensão do funcionamento do *hardware* utilizado, neste caso concreto, da componente rádio. Finalizados estes testes, ficaremos a conhecer algumas da suas limitações, podendo evitá-las, bem como explorar as potencialidades da sua utilização.

O procedimento foi repetido em 3 locais distintos, por forma a efectuar-se uma análise mais pormenorizada e correcta da comunicação rádio. Desta forma, optamos por escolher um local com muita movimentação (centro comercial), um espaço ao ar livre com árvores (parque de merendas) e um local mais pequeno, com paredes e vidros (empresa). Neste último, foi ainda utilizado a transferência de pacotes com e sem Bluetooth activo (existência de processos de *scan* que percorriam os canais do Bluetooth). O material necessário para se proceder a estes testes foi:

- 2 Módulos XBee série 2;
- 2 XBee Explorer USB;
- 2 Cabos Mini-USB;
- 2 Computadores portáteis;

Foi necessário proceder à configuração dos módulos XBee série 2. Para isso, instalou-se a ferramenta X-CTU e efectuou-se os passos apresentados nos apêndices A.1, A.2 e A.5.1 para configurar um ZC e os apêndices A.1, A.3 e A.5.1 para configurar um ZR. Como complemento, temos um exemplo na figura 4.1.

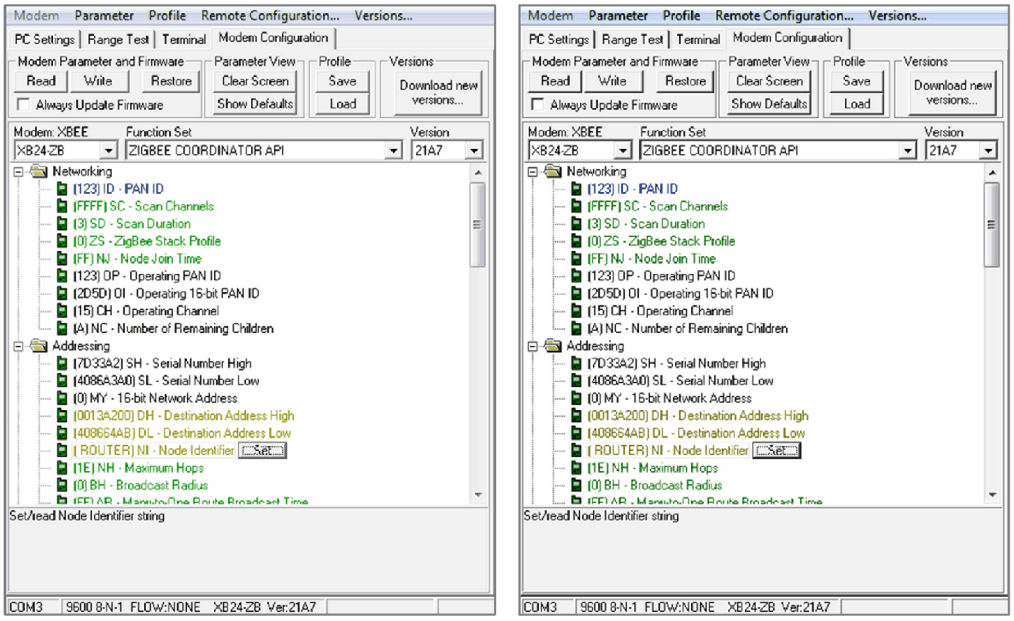

Figura 4.1: Configurações no ZC e ZR

Finalizadas estas configurações, temos uma rede ZigBee criada, com uma topologia em par. Depois foi necessário criar um transmissor[1] configurado como ZC e um receptor[1] de pacotes como ZR, com o recurso à XBee-API na linguagem de programação Java (ver secção 5.4.6). No apêndice C, é possível visualizar 2 excertos de código sobre o envio e detecção do estado do pacote transmitido.

Estes testes foram feitos em modo segurança activo e em linha de vista, sempre que possível (por vezes, mediante as distâncias, a linha de vista não foi possível devido à existência de objectos). O transmissor envia 250 pacotes em cada teste e o conteúdo do *payload* são caracteres alfanuméricos gerados pseudo-aleatoriamente. Cada teste é repetido 25 vezes para o tamanho do *payload* 24 *bytes* e posteriormente é repetido o mesmo procedimento para 64 *bytes*.

As limitações práticas levaram a fazer-se menos testes do que inicialmente foi pensado, nomeadamente a longa duração de cada teste, portanto as conclusões retiradas podem ser limitadas.

[1]Módulo XBee ligado a um XBee Explorer USB e estes ligados por mini-USB a um computador portátil

Como um dos objectivos é perceber qual o alcance dos módulos nos diversos locais, este processo foi repetido ainda para 4 distâncias diferentes, aproximadamente 15, 20, 25 e 30 metros. O transmissor não muda de posição durante os testes, apenas a posição do receptor é alterada. Relativamente à posição das antenas, esta não se altera.

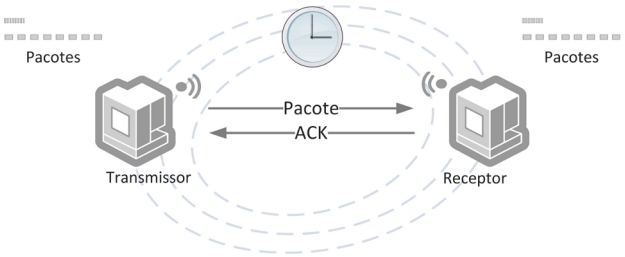

Figura 4.2: Esquema do envio de pacotes

Após o envio dum pacote, o transmissor aguarda que chegue o ACK a confirmar que este foi recebido por parte do receptor. É também registado o RSSI à chegada de cada pacote e o RTT de cada transmissão.

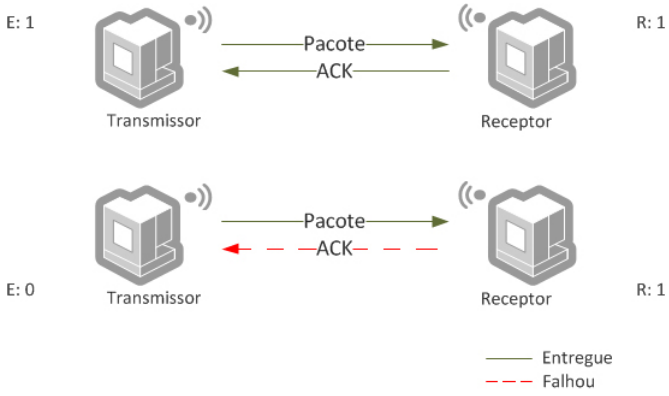

Figura 4.3: Possível falha durante os testes

Do lado oposto, o receptor regista o número de pacotes recebidos (taxa de recepção). Este procedimento é necessário para saber se os pacotes que o transmissor diz serem perdidos deve-se ao facto destes não chegarem ao receptor ou se o ACK da confirmação se perdeu. Podemos ver um exemplo prático na figura 4.3.

Neste caso concreto, como os testes foram feitos em cenários muito específicos e, por questões logísticas, não foram feitos muitos testes, não é de esperar que fazendo testes semelhantes se vão obter exactamente os mesmos resultados, pelo que estes são apenas indicativos.

De seguida, é possível visualizar os testes realizados descritos em gráficos, que estão divididos em 2 secções: testes de cobertura e testes de tempo.

4.1.2 Testes de Cobertura

Os testes de cobertura serviram para ter uma noção de qual a área coberta por cada módulo de rádio. Assim, como já foi dito, tínhamos um dispositivo que se mantinha no mesmo local ao longo dos testes (transmissor) e outro que se ia afastando (receptor).

Analisando a tabela 4.1, podemos verificar que a taxa de transmissão de pacotes para os quatro locais é bem distinta. Esta é maior no parque de merendas (com árvores), onde pode atingir os 25 metros.

A 15 metros, o parque, a empresa e o centro comercial têm uma taxa de transmissão de aproximadamente 100%. Esta distância é a única onde foi possível transmitir pacotes na empresa com Bluetooth activo (pelo que, a activação do Bluetooth provoca interferências na transmissão).

Tabela 4.1: Fiabilidade da transmissão a distâncias diferentes (24 *bytes*)

Locais	15	20	25	30
Parque	100%	96%	100%	-
Centro comercial	95%	100%	-	-
Empresa	99%	-	-	-
Empresa (Bluetooth activo)	2%	-	-	-

Para 20 metros, apenas no parque e no centro comercial houve transmissão, com uma taxa de aproximadamente 100%. O facto da transmissão de pacotes não ser possível na empresa (sem Bluetooth activo) é justificado com a existência de objectos que impediram a mesma, nomeadamente mesas, múltiplas paredes e vidros.

Por fim, a 25 metros, apenas se conseguiu transmitir no parque. Novamente, objectos, pessoas e interferências (redes sem fios) podem ter influenciado a transmissão no centro comercial.

A 30 metros, nestes locais, não foi possível obter resultados. Estes resultados verificaram-se independentemente do tamanho do pacote (ver tabela 4.2), onde as pequenas diferenças apresentadas são justificadas com interferências momentâneas que afectaram alguns testes.

Durante os testes realizados, foi possível detectar a falha que foi identificada na figura 4.3. Na tabela 4.3, temos um exemplo dessa falha, onde é facilmente identificável que o número de pacotes registados no transmissor em cada teste nem sempre coincide com

Tabela 4.2: Fiabilidade da transmissão a distâncias diferentes (64 *bytes*)

Locais	15	20	25	30
Parque	100%	100%	100%	-
Centro comercial	100%	100%	-	-
Empresa	99%	-	-	-
Empresa (Bluetooth activo)	5%	-	-	-

o número de pacotes registados no receptor. Este facto acontece porque a entrega do ACK não foi possível. Esta falha ocorreu apenas nos testes onde existiram problemas na comunicação, nomeadamente nos testes realizados na empresa com Bluetooth activo.

Tabela 4.3: Número de pacotes registados no transmissor *versus* número de pacotes registados no receptor na empresa com Bluetooth activo a 15 metros de distância (24 *bytes*)

Número do Teste	1	2	3	4	5	6	7	8	9	10	11	12	13
Transmissor	18	34	12	51	10	12	11	4	0	0	0	0	5
Receptor	23	35	14	59	11	15	12	4	0	0	0	0	7

Número do Teste	14	15	16	17	18	19	20	21	22	23	24	25
Transmissor	2	1	9	1	0	0	0	0	0	0	0	0
Receptor	3	2	11	1	0	0	0	0	0	0	0	1

Relativamente à intensidade de sinal dos pacotes (RSSI) recebidos pelo receptor, os resultados obtidos vieram comprovar que esta medida não nos garante que a taxa da transmissão de pacotes com sucesso seja a mais elevada. Como é visível na figura 4.4, o valor do RSSI obtido na transmissão de pacotes na empresa com Bluetooth activo foi superior ao valor registado sem Bluetooth activo (-90 dBm contra -94 dBm). No entanto, a taxa de transmissão sem Bluetooth activo foi substancialmente superior à do Bluetooth activo (99% contra 3%). Acontece a mesma situação nos resultados apresentados na figura 4.5 para pacotes de tamanho superior.

O facto de o RSSI ser maior com o Bluetooth activo resulta de a medição do RSSI ser feita apenas tendo em conta os pacotes recebidos — para um pacote ser recebido

tem que ter um valor mínimo de relação sinal/ruído, pelo que quando há mais ruído (Bluetooth activo) só são recebidos pacotes com maior RSSI.

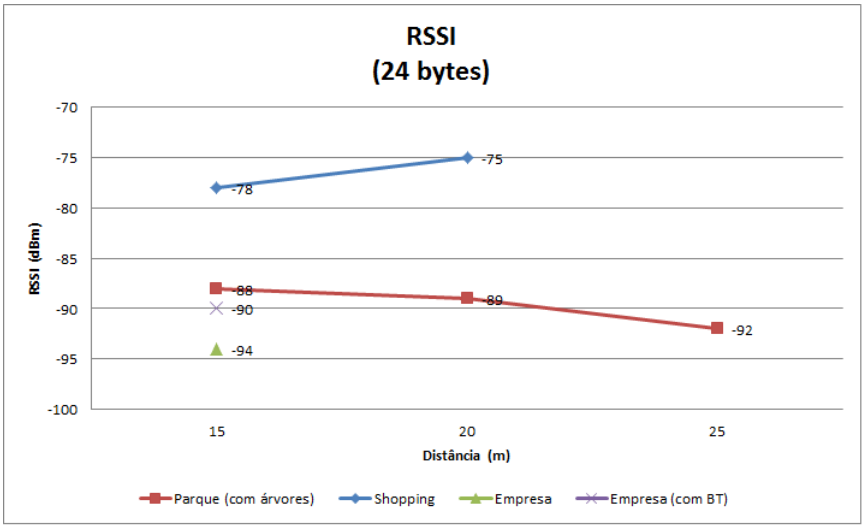

Figura 4.4: RSSI (24 *bytes*)

O aumento do RSSI para os pacotes de 24 *bytes* no centro comercial é causado pela existência de muitas pessoas a movimentar-se no local naquele instante (os testes foram realizados na zona da restauração e prolongaram-se no tempo, pelo que, na altura que foi feito o teste de menor distância existiam mais pessoas no local).

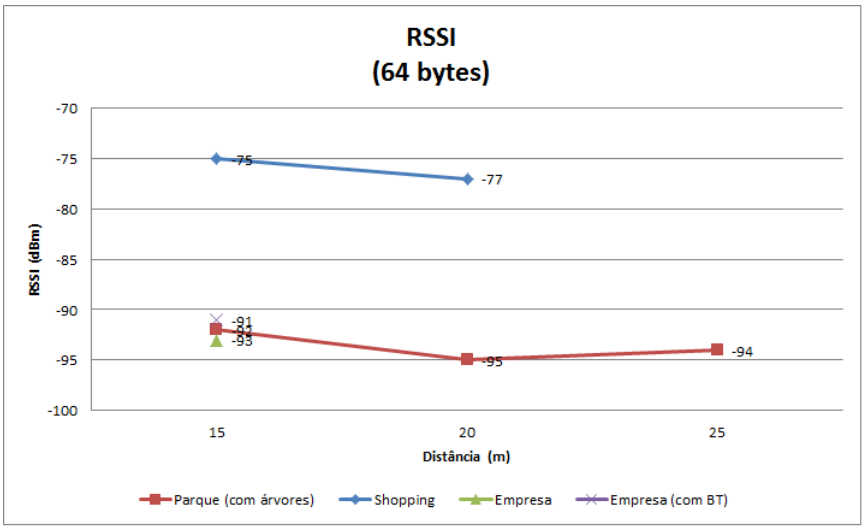

Figura 4.5: RSSI (64 *bytes*)

Outra conclusão retirada destes resultados foi que o RSSI não é proporcional à distância de transmissão. Por exemplo, ao analisar as figuras 4.4 e 4.5, poderia pensar-se

que para 25 metros iria existir transmissão no centro comercial, uma vez que, o valor do RSSI é de aproximadamente -75 dBm, sendo que nos outros locais o valor de RSSI está mais próximo do limite e mesmo assim existiu transmissão. No entanto, isto não se verificou.

4.1.3 Testes de Tempo

Os testes de tempo serviram para determinar o intervalo de tempo entre a chegada do pacote ao destino e o correspondente ACK (RTT). Os pacotes que foram perdidos não são contabilizados no tempo total.

O tempo médio do RTT para os testes realizados com 24 *bytes*, quando a taxa de transmissão é de 100%, varia entre os 88.4 e 95.7 ms (ver figura 4.6). À medida que a taxa de transmissão baixa, o RTT aumenta, devido ao facto das más condições de transmissão originarem a necessidade de efectuar retransmissões. Pode-se visualizar ainda que, o Bluetooth activo provoca interferências nas medições na empresa, fazendo não só diminuir a taxa de transmissão, como aumentar consideravelmente o RTT (1283.8 ms).

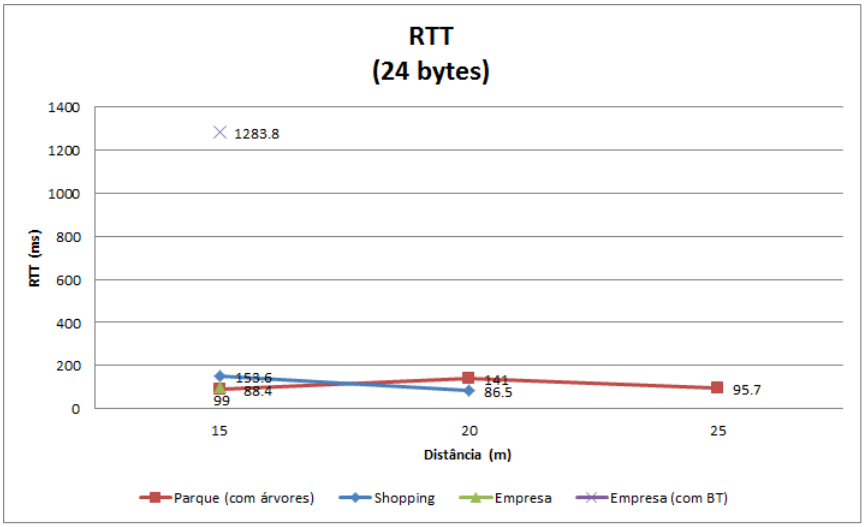

Figura 4.6: RTT (24 *bytes*)

Relativamente aos testes realizados com pacotes de tamanho 64 *bytes*, quando a taxa de transmissão é de 100%, o RTT varia entre 132 e 147 ms (ver figura 4.7). É visível um aumento no RTT comparativamente à situação anterior, justificado pelo facto do pacote ser maior, logo demorar mais tempo a ser transmitido. Em relação ao RTT registado na empresa com Bluetooth activo, este manteve-se elevado (1257.5 ms).

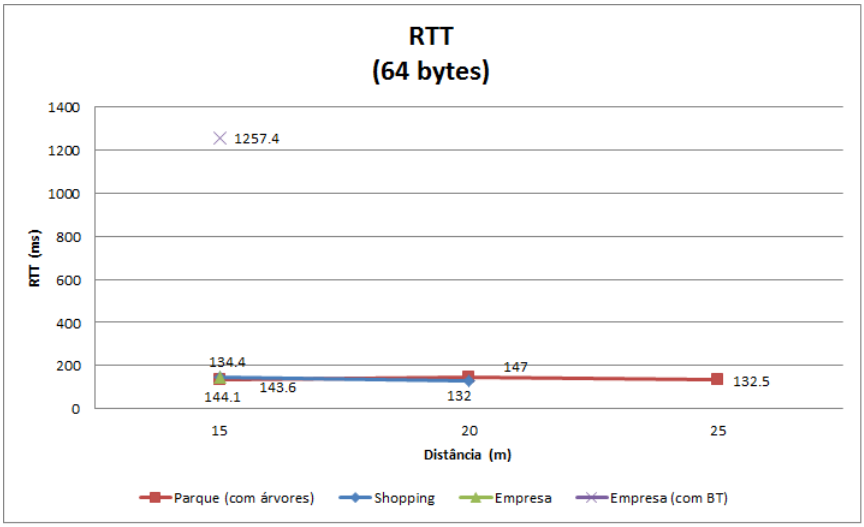

Figura 4.7: RTT (64 *bytes*)

4.2 Topologias de Rede

Os módulos XBee que são usados permitem todas as topologias apresentadas na secção 3.2.5 [14]. Para isto, basta decidir qual a topologia mais adequada à situação e efectuar a configuração no *software* X-CTU.

4.3 Encaminhamento Dinâmico

A inclusão de novos nós com o intuito de fazerem retransmissão de informação para outros nós da rede pode solucionar muitos problemas. No entanto, se não houver caminhos alternativos para colmatar eventuais falhas nos nós, parte da rede pode ficar inacessível. Esta diversidade permite que no caso de acontecer algum problema num dos dispositivos, o pacote continua a ser entregue no seu destino.

O seguinte teste, visível na figura 4.8, tem por objectivo perceber se os módulos XBee contemplam esta propriedade (encaminhamento dinâmico), fazendo com que a rede se adapte a circunstâncias anómalas.

Para atingir este objectivo, foi necessário um ZC, dois ZRs e um ZED. As configurações necessárias estão disponíveis nos apêndices A.1, A.2, A.3 e A.4. Quando a rede já se encontrava em funcionamento, desligou-se alternadamente os ZRs a fim de verificar se o ZED continuava a receber os pacotes vindos do ZC.

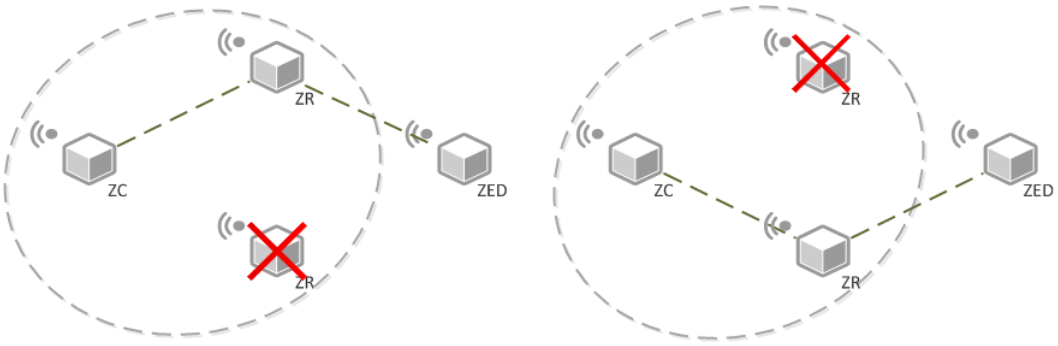

Figura 4.8: Encaminhamento Dinâmico

Concluiu-se que, independentemente do ZR que é desligado, o ZED continua a receber os pacotes destinados a ele. Existe um pequeno atraso até que a transferência de pacotes seja possível.

4.4 Segurança

Esta secção tem como fim aplicar as propriedades de segurança apresentadas na secção 3.2.8 nos módulos XBee, visto que de origem, esta característica não se encontra activa. A aplicação destas propriedades de segurança pode ser efectuada de duas formas diferentes: pré-configuração das chaves de ligação e obtenção das chaves durante a ligação à rede.

Para isto, são apresentadas as configurações necessárias nas sub-secções seguintes e detalhado todos os procedimentos de cada método. Assume-se que os dispositivos estão devidamente configurados.

4.4.1 Pré-Configuração das Chaves de Ligação

Para se proceder à pré-configuração das chaves de ligação nos módulos seguiram-se os procedimentos que se encontram descritos no apêndice A.5.1, com recurso ao *software* X-CTU (ver secção A.1). Posteriormente, é necessário verificar que o parâmetro Association Indication (AI) [2] retorna 0 em todos os dispositivos da rede. Se isto acontece, temos a rede formada.

Os parâmetros Encryption Enable (EE)[2], Extended PAN ID (ID)[2] e Link Key (KY)[2]

[2]Comando AT definido no apêndice B

são configurados em todos os dispositivos. Após a associação à rede segura, todas as transmissões de mensagens são cifradas com a chave pública. Ao definir o parâmetro Network Encryption Key (NK)[2] com 0 no ZC, foi gerada uma chave de rede aleatória. Como a chave de ligação (KY) foi configurada em todos os dispositivos, a chave de rede foi cifrada com essa chave e foi enviada aos dispositivos quando se ligaram.

4.4.2 Obtenção das Chaves durante a Ligação à Rede

Para se proceder à obtenção das chaves durante a ligação à rede nos módulos seguiram-se os procedimentos que se encontram no apêndice A.5.2, com recurso ao *software* X-CTU (ver secção A.1). Depois é necessário verificar que o parâmetro AI[2] retorna 0 em todos os dispositivos da rede. Se isto acontece, temos a rede formada.

Os parâmetros EE[2], ID[2] e KY[2] são configurados igualmente em todos os dispositivos. Ao definir o parâmetro NK[2] com 0 no ZC, foi gerada uma chave de rede aleatória. Como a chave de ligação (KY) foi configurada com 0, a chave de rede foi enviada sem qualquer mecanismo de segurança (em claro) quando os dispositivos se ligaram. Caso um atacante estivesse à escuta, poderia facilmente obter a chave de rede. Esta abordagem apresenta uma vulnerabilidade na rede e não é recomendada.

Capítulo 5

Descrição do Sistema

Neste capítulo é descrito o sistema implementado. Analisaremos as suas principais características, a sua arquitectura e as tecnologias utilizadas.

5.1 Principais Características

O sistema implementado contempla as seguintes características, que auxiliam o administrador na gestão e no controlo automático da rede de sensores:

- Acesso à rede controlado;
- Acesso à informação através de pedidos Hypertext Transfer Protocol Secure (HTTPS);
- Facilidade de configuração de novos dispositivos de rede;
- Possibilidade de inserção de novos sensores aos dispositivos de rede;
- Detecção e inserção no sistema de novos dispositivos e sensores de forma automática, com envio de *email* ao administrador;
- Monitorização e controlo dos dispositivos e sensores da rede;
- Aplicação de filtros na informação recolhida. São gerados alertas de aviso e, caso o administrador pretenda, envio de *email*;
- Garantia de integridade das mensagens e do sistema;

- Configuração local e remota de parâmetros AT[1];
- Estatísticas de envio e recepção de mensagens;
- Registo de avisos, *debug*, erros e pedidos em ficheiros *log*.

5.2 Visão Geral do Sistema

Na figura 5.1 é visível um esquema da rede de sensores, em que, à esquerda, temos o servidor central da empresa com acesso remoto ao DL, e este, por sua vez, tem acesso à rede de sensores instalados no estabelecimento (rede interna).

Comparativamente ao sistema apresentado na secção 2.2.3, existem melhorias ao nível da comunicação entre dispositivos, nomeadamente em termos de segurança e fiabilidade, e ainda, o acréscimo de serviços no DL.

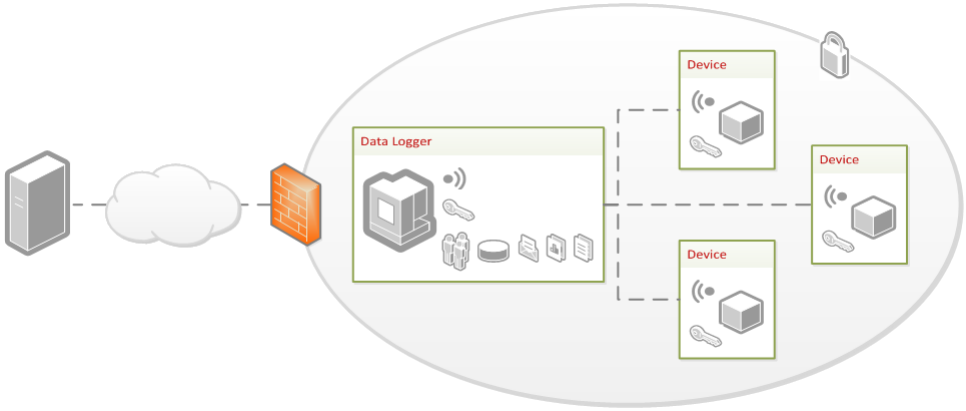

Figura 5.1: Visão Geral do Sistema

O DL recolhe e armazena toda a informação da rede numa base de dados local. É possível aceder a este dispositivo remotamente, com as devidas regras de segurança associadas, efectuando pedidos HTTPS para obter informações relativas à rede. As respostas vêm no formato JavaScript Object Notation (JSON). Neste dispositivo, são efectuados registos em ficheiros *log* do funcionamento do sistema.

Toda a informação recolhida pelo DL passa por um filtro dos valores recolhidos pelos sensores, com o intuito de perceber se estes estão dentro dos padrões normais. Estes parâmetros são definidos na instalação dos sensores e podem ser modificados facilmente

[1]Comandos utilizados para alterar configurações do *modem*

através do *dashboard*. Caso estes parâmetros não sejam cumpridos ou ocorram falhas, são gerados alertas e enviados ao administrador de rede.

Relativamente aos dispositivos de rede, estes têm a responsabilidade de comunicar com os seus sensores, guardar os valores recolhidos em memória e posteriormente enviar ao DL responsável. Os valores recolhidos são enviados periodicamente. A periodicidade é definida pelo administrador na instalação e pode ser modificada a qualquer momento através de pedidos HTTPS. Os dispositivos de rede não têm nenhum mecanismo de registo do seu funcionamento.

Sobre a comunicação entre os dispositivos e o DL, esta continua a ser feita através de portadoras rádio. No entanto, agora existem mecanismos de monitorização e controlo de mensagens. São definidos limiares de tempo no DL para cada dispositivo e sensor pertencentes à rede e em caso de incumprimento, é enviado um *keep alive*. Desta forma, o DL é capaz de perceber se um dispositivo ou sensor em concreto se encontra indisponível. Caso isto se verifique, o DL informa o administrador de rede via *email*. As mensagens que não sejam relativas a informações recolhidas pelos sensores são designadas por mensagens de controlo.

Foram introduzidos mecanismos de segurança associados à troca de mensagens. O conteúdo deixou de ser visível na rede, impedindo ataques de recolha passiva de informações. A integridade das mensagens também é garantida evitando ataques de modificação de mensagens.

Outra preocupação resolvida, foi a entrada de novos dispositivos na rede. Foram adicionados mecanismos de segurança de modo a que apenas dispositivos fidedignos (que conheçam a chave de rede) tenham acesso à rede. Estes mecanismos impossibilitam ainda a junção de dispositivos falsos à rede e a introdução de dados maliciosos.

Em relação ao *hardware*, os dispositivos que compõem a rede de sensores são constituídos por um Arduino Fio juntamente com um módulo XBee. Relativamente ao DL, o *hardware* utilizado é o DreamPlug.

5.3 Arquitectura do Sistema

Depois de analisar o sistema de forma muito superficial, vamos descrever com mais detalhe os processos existentes no DL e nos dispositivos de rede.

No DL temos um processo *multi-threading* (desempenha várias funções em simultâ-

neo) que tem como principais objectivos recolher, monitorizar, processar e guardar a informação recolhida pelos sensores instalados.

De um modo geral, este processo recebe todas as mensagens vindas dos dispositivos da rede, avalia-as, podendo mesmo aplicar novas medidas na rede ou avisar o técnico responsável da ocorrência de alguma anomalia, e regista-as no sistema para, se necessário, consultar mais tarde. Regista ainda tudo o que acontece na rede em ficheiros *log*. Para evitar o crescimento destes ficheiros, enchendo o espaço de armazenamento, foi criado um mecanismo de compressão diário. São guardados apenas os quatro ficheiros mais actuais.

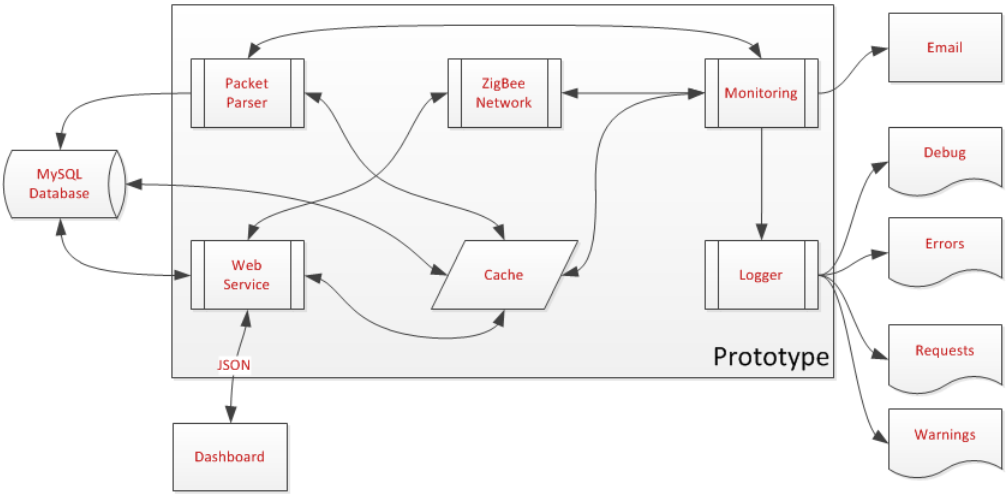

Figura 5.2: Arquitectura do Sistema

A fim de compreender melhor as funções deste processo, vamos dividi-lo em seis funções principais que se encontram na figura 5.2 destacadas sob um fundo cinzento. As setas representam as ligações lógicas entre funções e recursos:

***Packet Parser*:** Detecta e filtra todos os pacotes que têm como destino o coordenador da rede. Consoante o tipo de mensagem (controlo ou dados), desempenha uma tarefa específica. Actualiza, caso os dados em *cache* sejam diferentes, os parâmetros de rede com a chegada dos pacotes para esse dispositivo específico na base de dados e em *cache*. Finalmente adiciona o conteúdo do pacote recebido na base de dados (dados recolhidos pelo sensor do dispositivo).

Pedidos HTTPS: Responde a pedidos HTTPS externos ao sistema. Apenas utilizadores que conheçam as credenciais da API podem usufruir desta potencialidade. O método utilizado para este controlo é designado por *basic access authentication*

e permite efectuar pedidos HTTPS com autenticação, onde o utilizador e a *password* encontram-se no cabeçalho do pacote.

As respostas correspondem a informações recolhidas pela rede podendo ser ou não em tempo real. É ainda possível obter ou enviar parâmetros de rede aos dispositivos que constituem a rede (com recurso ao sub-processo *ZigBee Network*);

***Cache*:** Guarda em memória informações sobre os dispositivos e sensores da rede. Este armazenamento permite acesso mais rápido a dados que constantemente são necessários, evitando contínuas ligações à base de dados (o que tornaria o processo mais moroso e dispendioso em termos de recursos);

***ZigBee Network*:** Responsável por enviar mensagens com o intuito de obter ou guardar parâmetros da rede;

***Monitoring*:** Analisa, compreende e avalia o funcionamento e a recolha de dados na rede. Controla todos os dispositivos e sensores da rede, detectando se estes se encontram indisponíveis. Consegue detectar novos dispositivos ou sensores na rede e adiciona-os ao sistema de forma automática. Aplica filtros aos dados recebidos nas mensagens para saber se estão de acordo com o esperados. Em caso de anomalias, informa o administrador de rede;

***Logger*:** Regista o funcionamento, avisos, pedidos e erros ocorridos no sistema em ficheiros *log*.

Este processo necessita de alguns recursos externos, que servem de auxílio à gestão da rede. Esses recursos são:

Servidor de *email*: Utilizado pelo sistema para enviar *emails* ao administrador de rede, de forma a ajudar na detecção ou correcção de anomalias existentes.

Servidor de base de dados: Serve para armazenar a informação dos dispositivos e sensores da rede, incluindo parâmetros de rede e dados recolhidos. Podem ser acedidos ou actualizados a qualquer momento pelo sistema.

Sistema de ficheiros: Armazena no directório *log* o que o sub-processo *Logger* escreve. Esta informação fica disponível ao administrador de rede e serve para analisar e corrigir anomalias detectadas. Através deste directório, é ainda possível fazer uma análise estatística sobre o funcionamento da rede;

Em termos de aplicações, podemos aceder à informação através de um *dashboard* (visualização gráfica do conteúdo recolhido), que utiliza os métodos HTTPS criados, permitindo desta forma uma melhor interacção entre o sistema e o utilizador final.

A comunicação entre o DL e o módulo XBee é estabelecida através da utilização da XBee-API (ver secção 5.4.6).

Relativamente aos dispositivos de rede, o processo é simples comparativamente ao que se encontra presente no DL. Este processo tem como principais objectivos recolher as informações dos seus sensores e enviá-las para o DL. Têm em comum a função *Packet Parser* presente no DL. No entanto, as mensagens recebidas apenas desempenham operações simples.

A comunicação entre o Arduino e o módulo XBee é estabelecida através da utilização da Application Programming Interface (API) XBee-Arduino (ver secção 5.4.7).

5.4 Tecnologias Utilizadas

Nesta secção são apresentadas todas as tecnologias utilizadas pelo sistema, com uma explicação sobre essas ferramentas, bem como a razão que levou a utilizá-las, assumindo a importância de serem *software open-source*.

Nas sub-secções 5.4.1, 5.4.2 e 5.4.3 temos as tecnologias que se encontram instaladas no DL. Na primeira temos o sistema operativo utilizado, depois a base de dados local onde é armazenada toda a informação e, por fim, as tecnologias usadas para servir o *dashboard* e o *web service*.

Nas duas seguintes, 5.4.4 e 5.4.5 respectivamente, temos ferramentas desenvolvidas pelas empresas competentes no intuito de ajudar na programação do Arduino e na configuração dos módulos XBee.

Finalmente, nas sub-secções 5.4.6 e 5.4.7, temos duas bibliotecas que permitem a conectividade entre XBee-DL e XBee-Arduino.

5.4.1 Sistema Operativo

O sistema operativo utilizado no DL foi a distribuição Linux Ubuntu, versão 9.04 Advanced RISC Machine (ARM) de 32 *bits* disponibilizada pela DreamPlug. Em [13],

temos todos os sistemas operativos disponibilizados para este *hardware*.

5.4.2 MySQL

O MySQL é um sistema de gestão de bases de dados (SGBD) relacionais, que utiliza a linguagem Structured Query Language (SQL) e usado normalmente em aplicações que necessitam de organização de dados. Este *software* é *open-source* e possui características como alto desempenho, confiabilidade e facilidade de uso.

A escolha desta tecnologia deveu-se à necessidade de utilizar uma ferramenta que possibilitasse o armazenamento de dados e a interacção com a linguagem de programação Java (através do *driver* JDBC). Foi também escolhida por ter familiaridade com este.

5.4.3 Apache e Apache Tomcat

A utilização do servidor Apache neste projecto é justificada com a necessidade de um servidor HTTPS para servir a aplicação *web* (*dashboard*) desenvolvida na linguagem de programação PHP.

O projecto *Apache Tomcat* é um servidor *open-source* que é utilizado em aplicações desenvolvidas na linguagem de programação Java para *web*. Como o *web server* desenvolvido foi na linguagem Java, foi necessário utilizá-lo.

5.4.4 Arduino IDE

O Arduino IDE é um *software open-source* desenvolvido pela *Arduino Team*. Encontra-se disponível para os sistemas operativos Windows, Linux e MAC OS na versão 1.0.1 [27].

A principal função desta aplicação é desenvolver e efectuar *upload* de código (normalmente designado por *sketch*) no *hardware* disponibilizado pela empresa. Este *software* permite ainda verificar erros de sintaxe antes de uma submissão e saber se esta foi bem sucedida. A conexão ao *hardware* é estabelecida através de porta série [25].

O motivo pelo qual esta ferramenta foi usada deveu-se ao facto da necessidade de um ambiente de programação, para criar o processo existente no Arduino, que recolhe informações dos sensores e envia-as para o DL. A facilidade com que é feito um *upload*

tornou-se também numa mais valia.

5.4.5 X-CTU

O X-CTU é um *software open-source* desenvolvido pela *Digi International Inc.* Encontra-se disponível apenas para o sistema operativo Windows na versão 5.2.7.5 [5]. Apesar disto, através da ferramenta Wine, é possível utilizá-lo nas plataformas Linux.

Este programa possui várias ferramentas de configuração e programação para os módulos XBee através de porta de série. É possível fazer *upgrade* de *firmware*, actualizar parâmetros de rede e verificar o desempenho da rede através do ambiente gráfico [5].

Este *software* é composto por quatro painéis principais [5]:

- ***PC Settings:*** Permite a ligação à porta de série onde o módulo XBee que se encontra ligado;
- ***Range Test:*** Permite testar o alcance entre dois dispositivos, nomeadamente obter a taxa de transmissão e a potência média do sinal recebido;
- ***Terminal:*** Permite aceder à porta série emulada, possibilitando o envio de comandos AT para dispositivos na rede ZigBee;
- ***Modem Configuration:*** Permite verificar ou modificar o *firmware* carregado, assim como configurar parâmetros de rede.

A razão da escolha desta ferramenta prende-se ao facto de esta ser extremamente útil, nomeadamente para carregar e actualizar o *firmware* a usar, por meios simplificados.

5.4.6 XBee-API

A XBee-API é uma biblioteca desenvolvida para as plataformas Windows, Mac OS e Linux na linguagem de programação Java que possibilita a comunicação, no modo API, com os módulos XBee/XBee-Pro para a série 1 (802.15.4) e série 2 (ZNet 2,5 e ZB/ZigBee Pro)[2], desde que as plataformas suportem a linguagem Java na versão 5 ou superior e RXTX [22].

[2]Requer o modo API, definindo o parâmetro AP=2. Nos casos em que usamos a série 2, é necessário instalar o *firmware* API e em seguida, definir o parâmetro AP=2. Não funciona correctamente com o parâmetro AP=1 [22, 23].

Esta biblioteca foi escolhida porque possibilita a comunicação entre o módulo XBee e o DL.

5.4.7 XBee-Arduino

A XBee-Arduino é uma biblioteca desenvolvida para Arduino que possibilita a comunicação, no modo API, com os módulos *XBee* para a série 1 (802.15.4) e série 2 (ZB Pro/ZNet)[2].

Inclui suporte para a maioria dos pacotes, incluindo *TX/RX*, *AT Command*, *Remote AT*, *I/O Samples* e *Modem Status* [23].

Decidiu-se usar esta biblioteca porque permite estabelecer comunicação entre o módulo XBee e o Arduino nos dispositivo de rede.

Capítulo 6

Desenvolvimento do Sistema

Neste capítulo são descritos todos os pormenores técnicos utilizados na implementação do sistema. Inicialmente, são apresentadas as configurações para os diferentes tipos de dispositivos na rede ZigBee. Depois, é mostrado a forma como todas as funções, já apresentadas, interagem umas com as outras no *web server* desenvolvido, que serve de ligação entre o DL e o ZC, assim como visualizar todas as funcionalidades do *dashboard* que actua directamente com o mesmo. Por fim, é descrito o *sketch* presente nos dispositivos de rede.

6.1 Configuração da Rede ZigBee

Como já foi referido, a distribuição de dispositivos numa rede ZigBee pode ter várias topologias (par, estrela, malha ou árvore). Esta característica deve-se à existência de 3 tipos de dispositivos: ZC, ZR e ZED.

Como não existe uma topologia certa, para a construção deste protótipo foram realizados alguns testes em diferentes cenários utilizando as topologias em par e em malha. Sempre que a capacidade da componente rádio era suficiente para comunicar com um outro módulo ZigBee, era utilizado a topologia em par. Caso existisse muitas falhas na transmissão de mensagens ou, no pior caso, não houvesse comunicação entre módulos, eram acrescentados novos dispositivos capazes de retransmitir as mensagens enviadas, tornando a comunicação possível ou em melhores condições.

De seguida são apresentadas, de forma genérica, as configurações necessárias para cada tipo de dispositivo.

Para configurar o ZC da rede é necessário proceder aos seguintes passos:

1. Ligação ao módulo XBee (Apêndice A.1)
2. Configurar ZC modo API (Apêndice A.2)
3. Activar modo segurança no ZC (Apêndice A.5.1 ou A.5.2)[1]

Caso pretendamos configurar um ZR, são:

1. Ligação ao módulo XBee (Apêndice A.1)
2. Configurar ZR (Apêndice A.3)
3. Activar modo segurança no ZR (Apêndice A.5.1 ou A.5.2)[1]

E finalmente, para configurar um ZED, são:

1. Ligação ao módulo XBee (Apêndice A.1)
2. Configurar ZED (Apêndice A.4)
3. Activar modo segurança no ZED (Apêndice A.5.1 ou A.5.2)[1]

6.2 *Web Server*

Como já foi referido no capítulo anterior, este processo encontra-se instalado no DL e serve de apoio ao sistema responsável pela monitorização, recolha, acesso e registo da rede de sensores instalada.

6.2.1 Construção da Base de Dados

O sistema de base de dados serve para armazenar toda a informação relativa à rede, bem como a informação recolhida pelos sensores, para posteriormente ser analisada e tratada. Assim, todos as informações relativas aos dispositivos e sensores da rede encontram-se salvaguardados.

[1]Dependendo do tipo de segurança que se quer aplicar

6.2.1.1 Modelo Relacional

O modelo relacional (ER) é um modelo de dados que se baseia em dois conceitos: entidade e relação. Uma entidade é um elemento caracterizado pelos dados que são recolhidos na sua identificação, vulgarmente designado por tabela. Na construção da tabela identificam-se os dados da entidade. A atribuição de valores a uma entidade constrói um registo da tabela. A relação determina o modo como cada registo de cada tabela se associa a registos de outras tabelas.

Neste sentido, foi elaborado o modelo relacional, visível na figura 6.1, para posteriormente se proceder à construção da base de dados.

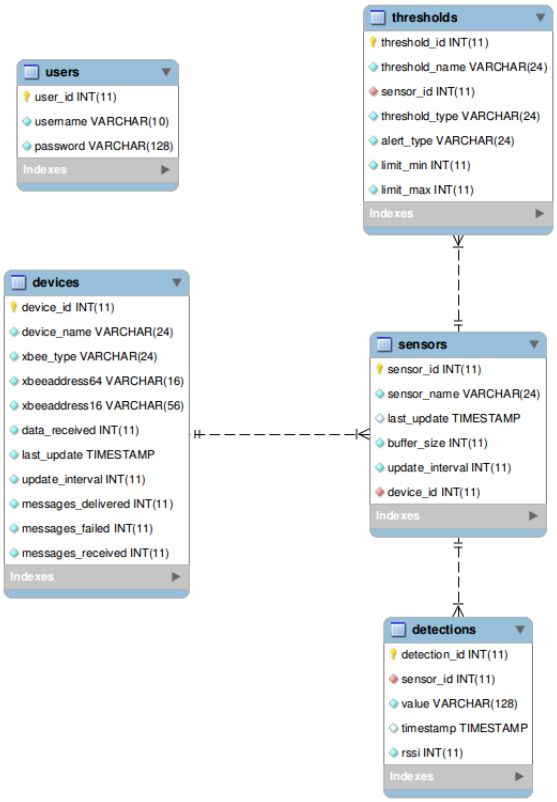

Figura 6.1: Modelo Relacional

Tendo em vista o modelo relacional apresentado, foram criadas as seguintes tabelas:

users - Contém todos os utilizadores e respectivos dados confidenciais que têm acesso ao *dashboard*. Apenas utilizadores com as devidas permissões de acesso/escrita podem aceder às informações da rede. Relativamente ao armazenamento das *passwords* dos utilizadores na base de dados, apenas é guardado o seu *hash*;

sensors - Informação sobre todos os sensores que se encontram agregados aos dispositivos de rede. Esta informação deve estar sincronizada e actualizada com a rede actual, isto é, só deve conter os sensores que se encontram em funcionamento. Um dispositivo pode ter vários sensores, no entanto, um sensor apenas pode estar ligado a um dispositivo;

devices - Guarda informação sobre todos os dispositivos da rede e respectivos dados. Esta informação deve estar sincronizada e actualizada com a rede actual, isto é, só deve conter os dispositivos que se encontram em funcionamento. Cada coordenador da rede é responsável por um conjunto de dispositivos. Regista ainda a taxa de transmissão e recepção de mensagens de cada dispositivo;

detections - Regista a informação recolhida pelos sensores da rede;

thresholds - Limiares associados a cada sensor presente no sistema. Os limiares podem ter níveis de prioridade diferentes, assim como desempenhar funções distintas. A detecção de valores fora destes limiares despoleta acções no processo de monitorização.

6.2.2 Mensagens

Foram utilizadas duas APIs, já descritas anteriormente, que possibilitam a comunicação com os módulos rádio, enviando pacotes de um lado para o outro. Interessa agora perceber de que forma é feita a transmissão de dados.

Na figura 6.2 temos a estrutura dos pacotes no modo API. O primeiro *byte* indica o inicio de um pacote (0x7E), depois os 2 *bytes* seguintes indicam o tamanho do pacote, isto é, o número de *bytes* entre o tamanho e o *checksum*. De seguida, temos o conteúdo do pacote. Finalmente, o último *byte* indica o *checksum* calculado pelo nó que originou o pacote, permitindo verificar a integridade do pacote.

Existem vários tipos de pacotes definidos na API. Na tabela 6.1, temos os tipos que foram utilizados na troca de mensagens. Desta forma é possível distinguir os pacotes recebidos, encaminhando-os para a função a desempenhar após a recepção.

Sempre que chega uma nova mensagem, é actualizada uma variável que regista o *timestamp* da última mensagem recebida para esse dispositivo ou sensor. Depois, periodicamente, existe uma *thread* que verifica quais os dispositivos/sensores que ultrapassaram o limite de tempo sem comunicar com o agregador (este limite é estipulado

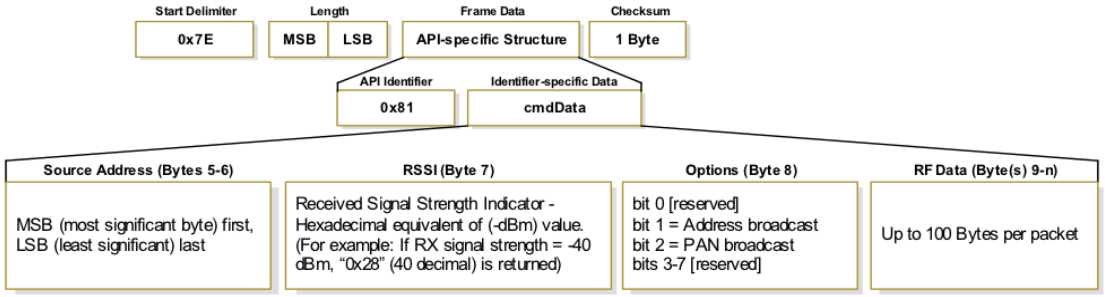

Figura 6.2: Estrutura de um Pacote no Modo API (de [1])

pelo administrador durante a instalação e pode ser modificado a qualquer momento no *dashboard*). Caso este acontecimento ocorra, é enviado um *keep-alive* para o dispositivo em concreto, para que o agregador saiba o que ocorreu com o dispositivo. Assim, através deste método é ainda possível saber o estado de todos os dispositivos e sensores que constituem a rede.

Tabela 6.1: *API Frame Names*

API Frame Names	API ID
AT Command	0x08
ZigBee Transmit Request	0x10
Remote Command Request	0x17
AT Command Response	0x88
ZigBee Transmit Status	0x8B
ZigBee Receive Packet (AO=0)	0x90
Remote Command Response	0x97

Para uma melhor organização das mensagens trocadas, foram criados dois tipos de mensagens: controlo e dados.

6.2.2.1 Controlo

As mensagens de controlo, tal como o nome indica, servem para controlar o que acontece na rede. O coordenador pode a qualquer momento necessitar de uma informação e pedir especificamente a um dispositivo que lhe indique o que pretende. De seguida, temos a estrutura e um exemplo de uma mensagem enviada de um dispositivo de rede para o coordenador:

Listing 6.1: Mensagens de Controlo

```
Formato: id_dispositivo|id_mensagem|tipo|valor
Exemplo: SALAO|0|1|ZR
```

Descrição:

- **id_dispositivo:** Identifica o dispositivo e é atribuído pelo administrador na instalação
- **id_mensagem:** 0
- **tipo:**
  0 - *Keep Alive*
  1 - Tipo de dispositivo
- **valor:**
  0 - Desligado ou 1 - Ligado
  ZR ou ZED

Neste caso em concreto, o coordenador da rede recebeu um pacote vindo do dispositivo de rede "SALAO", é uma mensagem de controlo e indica que o dispositivo em questão é um ZR.

6.2.2.2 Dados

As mensagens de dados servem para enviar os dados que são recolhidos pelos sensores. De seguida, temos a estrutura e um exemplo de uma mensagem enviada de um dispositivo de rede para o coordenador:

Listing 6.2: Mensagens de Dados

```
Formato: id_dispositivo|id_mensagem|id_sensor|valor
Exemplo: SALAO|1|TEMPERATURA|20
```

Descrição:

- **id_dispositivo:** Identifica o dispositivo e é atribuído pelo administrador na instalação
- **id_mensagem:** 1

- **id_sensor:** Atribuído pelo administrador na instalação
- **valor:** Valor recolhido pelo sensor

Novamente, neste caso especifico, o coordenador da rede recebeu um pacote vindo do dispositivo de rede "SALAO", é uma mensagem de dados e indica que o sensor "TEMPERATURA" recolheu o valor 20ºC.

6.2.2.3 Comandos AT

Os comandos AT têm um papel diferente dos apresentados nas secções anteriores. Enquanto as mensagens de dados e de controlo referem-se especificamente à aplicação, estes servem para visualizar ou modificar parâmetros locais ou remotos na rede ZigBee.

No apêndice B.1, são apresentados todos os comandos que o sistema aceita juntamente com a sua descrição. Existiu uma filtragem dos comandos AT, pois eram demasiados e a maior parte não se adequavam ao problema. No entanto, facilmente se pode expandir o sistema para aceitar novos comandos.

6.2.3 Ficheiros de *Log*

Os ficheiros de *log* servem para registar o funcionamento do sistema. Estão divididos em quatro secções: avisos, erros, pedidos e rede.

Os ficheiros de aviso guardam todos *warnings* gerados pelo sistema, nomeadamente, quando os limiares não são cumpridos. Os ficheiros de erro contêm os erros que ocorreram durante o funcionamento e na medida do possível, com os factores que causaram esse erro. Relativamente aos ficheiros de pedidos, estes possuem todos os pedidos HTTPS efectuados ao sistema. Por fim, temos os ficheiros de rede, que documentam todas as mensagens trocadas pela rede. De seguida, temos dois exemplos do registo que é feito nos ficheiros de rede.

Listing 6.3: Recepção de um pacote vindo de um dispositivo

```
Received packet from XBee: apiId=ZNET_RX_RESPONSE(0x90),
        length=32,
        checksum=0x54,
        error=false,
        remoteAddress64=0x00,0x13,0xa2,0x00,0x40,0x86,0x64,0xab,
        remoteAddress16=0xc7,0x73,
```

```
option=PACKET_ACKNOWLEDGED,
data=0x30,0x31,0x7c,0x41,0x4b,0x7c,0x52,0x4f,0x55,
        0x54,0x45,0x52,0x7c,0x54,0x45,0x4d,0x50,0x7c,0x32,0x30
```

Neste caso, o coordenador recebeu um pacote, com origem de 0x0013a200408664ab (endereço 64 de *bits* do módulo XBee do dispositivo), de tamanho 32 *bytes* (soma dos campos *error* (1 *byte*), do *remoteAddress64* (8 *bytes*), do *remoteAddress16* (2 *bytes*), do *option* (1 *byte*) e do *data* (20 *bytes*)), e o conteúdo do pacote encontra-se em hexadecimal. Não ocorreu qualquer erro na entrega deste e este requer que seja enviado um ACK de confirmação.

Listing 6.4: Envio de um comando remotamente

```
Sending request to XBee: apiId=REMOTE_AT_REQUEST (0x17),
        frameId=1,
        command=DB,
        value=null,
        remoteAddr64=0x00,0x13,0xa2,0x00,0x40,0x86,0x64,0xab,
        remoteAddr16=0xff,0xfe,
        applyChanges=false

Received packet from XBee: command=DB,
        status=OK,
        value=0x34,
        apiId=REMOTE_AT_RESPONSE (0x97),
        length=16,
        checksum=0xe9,
        error=false,
        frameId=0x01,
        remoteAddress64=0x00,0x13,0xa2,0x00,0x40,0x86,0x64,0xab,
        remoteAddress16=0xc7,0x73
```

Aqui temos o envio de um comando remoto e respectiva resposta para o dispositivo 0x0013a200408664ab, com o intuito de saber o RSSI do último pacote que foi entregue por parte deste (parâmetro DB).

6.2.4 Limiares

A inclusão de limiares tem como objectivo controlar os valores recolhidos pelos sensores. Assim, cada sensor (caso se aplique) tem associado a si um intervalo de valores que são considerados normais. Caso isto não aconteça, é gerado um alerta e, caso o administrador pretenda, enviado um *email* para a sua caixa de correio.

Imaginando que temos um sensor de temperatura junto uma sala de servidores, podemos considerar então os seguintes intervalos:

- Valores inferiores a 10ºC ou superiores a 25ºC - **Função:** Enviar *email*
- Valores entre 10 e 25ºC - **Função:** Nenhuma (Valores normais)

```
jmmg@ubuntu:/opt/eclipse$ curl -vv "http://APIKey:123456@localhost:8080/NetworkWebServer/DemoServlet?f=getsensors&devid=ak&offset=0&limit=1"
* About to connect() to localhost port 8080 (#0)
*   Trying 127.0.0.1... connected
* Server auth using Basic with user 'APIKey'
> GET /NetworkWebServer/DemoServlet?f=getsensors&devid=ak&offset=0&limit=1 HTTP/1.1
> Authorization: Basic QVBJS2V5OjEyMzQ1Ng==
> User-Agent: curl/7.22.0 (i686-pc-linux-gnu) libcurl/7.22.0 OpenSSL/1.0.1 zlib/1.2.3.4 libidn/1.23 librtmp/2.3
> Host: localhost:8080
> Accept: */*
>
< HTTP/1.1 200 OK
< Server: Apache-Coyote/1.1
< Cache-Control: no-cache
< Access-Control-Allow-Origin: *
< Access-Control-Allow-Headers: Accept,Content-Type
< Content-Type: application/json;charset=UTF-8
< Transfer-Encoding: chunked
< Date: Fri, 28 Sep 2012 10:27:02 GMT
<
{"result":[{"sensorIdentifier":"TEMP","deviceIdentifier":"AK","bufferSize":1024,"lastUpdate":1348828011563,"thresholds":{"meddium":{"thresholdIde
ntifier":"limiar","deviceIdentifier":"AK","sensorIdentifier":"TEMP","thresholdType":"meddium","alertType":"email","limitMin":10,"limitMax":20}},"
updateInterval":3600,"available":true}]}
* Connection #0 to host localhost left intact
* Closing connection #0
```

Figura 6.3: Exemplo de um limiar associado a um sensor de temperatura

6.2.5 Pedidos HTTPS

A utilização dos pedidos HTTPS permite ao utilizador obter informações ou desempenhar funções na rede em tempo real. O acesso a estes pedidos é controlado, isto é, para usufruir destes pedidos é necessário conhecer as credenciais da API. Caso contrário, o acesso é negado. O método utilizado para este controlo é designado por *basic access authentication* e permite efectuar pedidos HTTPS com autenticação, onde o utilizador e a *password* encontram-se no cabeçalho do pacote. O canal utilizado para o estabelecimento de sessões deve ser seguro (HTTPS), caso contrário as credenciais podem ser capturados por um *sniffer* de pacotes (p.e. *wireshark* [10]). Relativamente às respostas obtidas, vêm no formato JSON.

De seguida, é especificado o formato genérico dos pedidos existentes:

Listing 6.5: Formato do pedido GET

```
http://url:porta/projecto/servlet?func&param_func
```

Descrição:

- **url:** Endereço Internet Protocol (IP) ou DNS do DL

- **porta:** Identificação da porta do servidor *Tomcat*. Por defeito é a 8080
- **projecto:** Nome do projecto
- **servlet:** Identificação do *servlet*
- **func:** Identificação da função a desempenhar
- **param_func:** Parâmetros necessários para desempenhar a função

```
jmmg@ubuntu:/opt/eclipse$ curl -vv "http://APIKey:123456@localhost:8080/NetworkWebServer/DemoServlet?f=getsensorssize&devid=ak"
* About to connect() to localhost port 8080 (#0)
*   Trying 127.0.0.1... connected
* Server auth using Basic with user 'APIKey'
> GET /NetworkWebServer/DemoServlet?f=getsensorssize&devid=ak HTTP/1.1
> Authorization: Basic QVBJS2V5OjEyMzQ1Ng==
> User-Agent: curl/7.22.0 (i686-pc-linux-gnu) libcurl/7.22.0 OpenSSL/1.0.1 zlib/1.2.3.4 libidn/1.23 librtmp/2.3
> Host: localhost:8080
> Accept: */*
>
< HTTP/1.1 200 OK
< Server: Apache-Coyote/1.1
< Cache-Control: no-cache
< Access-Control-Allow-Origin: *
< Access-Control-Allow-Headers: Accept,Content-Type
< Content-Type: application/json;charset=UTF-8
< Transfer-Encoding: chunked
< Date: Fri, 28 Sep 2012 10:29:18 GMT
<
{"result":1}
* Connection #0 to host localhost left intact
* Closing connection #0
```

Figura 6.4: Exemplo de um pedido HTTPS, pedindo o número de sensores que um dispositivo contém

Os métodos existentes no sistema, encontram-se divididos em quatro categorias distintas: dispositivos de rede, sensores dos dispositivos de rede, configuração de rede e, finalmente, acesso aos ficheiros *log*.

Para cada método, é feita uma descrição e apresentada numa tabela a forma como pode ser efectuado o pedido[2].

Os métodos relativos aos dispositivos da rede são (ver tabela 6.2):

Número de dispositivos - Número de dispositivos geridos pelo coordenador da rede. Pode ainda ser obtida toda a informação relativa ao dispositivo de rede.

Número de sensores - Número de sensores agregados a um dispositivo da rede específico. Pode ainda ser obtida toda a informação relativa a todos os sensores do mesmo.

[2]Apenas são apresentados os itens *func* e *param_func*. Os outros são iguais em todos os métodos

Intervalo de tempo - Tempo máximo (em segundos) em que o dispositivo pode estar sem comunicar com o coordenador da rede. Após ultrapassar este limite, são tomadas medidas para se apurar o porquê deste tempo ser excedido. Este limite pode ser alterado em qualquer instante.

Última actualização - Registo do *timestamp* da última mensagem recebida do dispositivo de rede pelo ZC. O *timestamp* actual menos o *timestamp* da última actualização nunca deve ser superior ao tempo máximo (intervalo de tempo).

Dados recolhidos (*bytes*) - Número de *bytes* de dados transferidos pelo dispositivo de rede.

Tipo - Retorna qual a função que o dispositivo desempenha na rede (ZC, ZR ou ZED).

Número de mensagens entregues - Regista o número de mensagens entregues com sucesso de um determinado dispositivo de rede.

Número de mensagens recebidas - Regista o número de mensagens recebidas com sucesso com origem de um dispositivo de rede.

Número de mensagens enviadas - Regista o número de mensagens enviadas para um dispositivo de rede específico. A subtracção deste valor com o número de mensagens entregues permite saber a taxa de transferência.

RSSI médio das mensagens - Retorna o RSSI médio de todas as mensagens bem sucedidas. É possível ter uma percepção da distância do ZC ao dispositivo de rede.

Remover - Permite remover um dispositivo e todos os sensores associados ao mesmo.

Procurar - Permite procurar dispositivos que contenham aquela *substring*.

Os métodos relativos aos sensores dos dispositivos da rede são (ver tabela 6.3):

Intervalo de tempo - Tempo máximo (em segundos) em que o sensor, através do dispositivo, pode estar sem comunicar com o coordenador da rede. Após ultrapassar este limite, são tomadas medidas para se apurar o porquê deste tempo ser excedido. Este limite pode ser alterado em qualquer instante.

Tabela 6.2: Métodos relativos aos dispositivos da rede

Tipo	Características	Função e Parâmetros
GET	Nº de dispositivos	getsize&hw=device
	Informação dos dispositivos	getdevices&s=0&e=2
	Nº de sensores	getsize&hw=sensor
	Informação dos sensores	getsensors&devid=0013&s=0&e=2
	Intervalo de tempo	getupdateinterval&devid=0013
	Última actualização	getlastupdate&devid=0013
	Dados recolhidos (*bytes*)	getdatareceived&devid=0013
	Tipo	gettype&devid=0013
	Nº de mensagens entregues	getmessagesdelivered&devid=0013
	Nº de mensagens enviadas	getmessagessent&devid=0013
	Nº de mensagens recebidas	getmessagesreceived&devid=0013
	RSSI médio das mensagens	getrssiavg&devid=0013
	Procurar	searchdevice&str=0013
SET	Intervalo de tempo	setupdateinterval&devid=0013&int=100
REMOVE	Remover	removedevice&devid=0013

Última actualização - Registo do *timestamp* da última mensagem recebida do dispositivo de rede pelo ZC. O *timestamp* actual menos o *timestamp* da última actualização nunca deve ser superior ao tempo máximo (intervalo de tempo).

Tamanho do *buffer* - Tamanho reservado para a recolha de dados no dispositivo. Este limite serve para não exceder a memória do dispositivo.

Dados recolhidos (*bytes*) - Registo do número de *bytes* de todas as mensagens de dados recebidas relativas a um sensor específico.

Remover - Permite remover um sensor de um dispositivo de rede.

Procurar - Permite procurar sensores que contenham aquela *substring*.

Tabela 6.3: Métodos relativos aos sensores da rede

Tipo	Características	Função e Parâmetros
GET	Intervalo de tempo	getupdateinterval&devid=0013&sensid=temp
	Última actualização	getlastupdate&devid=0013&sensid=temp
	Tamanho do buffer	getsize&hw=sensor
	Dados recolhidos	getbuffersize&devid=0013&sensid=temp
	Procurar	searchsensor&str=temp
SET	Intervalo de tempo	setupdateinterval&devid=0013&sensid=temp&int=10
	Tamanho do *buffer*	setbuffersize&devid=0013&sensid=temp&buffer=10
REMOVE	Remover	removedevice&sensid=temp

Relativamente aos métodos para efectuar a configuração da rede durante a execução do sistema, visíveis na tabela 6.4, é possível enviar e receber comandos AT para o coordenador (localmente) ou para os dispositivos de rede (remotamente).

Por fim, temos os métodos relativos aos ficheiros de *log* presentes (avisos, *debug*, erros, pedidos e de rede), visíveis na tabela 6.5, em que é possível estabelecer um número limite de linhas apresentadas.

Tabela 6.4: Métodos relativos aos sensores da rede

Tipo	Características	Função e Parâmetros
GET	Remoto	getremoteatcmd&devid=0013&cmd=db
	Local	getlocalatcmd&cmd=db
SET	Remoto	setremoteatcmd&devid=0013&cmd=ni&payload=salao
	Local	setlocalatcmd&cmd=ni&payload=routerk

Tabela 6.5: Métodos relativos aos ficheiros de *log*

Ficheiro	Função e Parâmetros
Registo de erros	getdebuglog getdebuglog&limit=10
Registo de debug	geterrlog geterrlog&limit=10
Registo dos pedidos	getrequestlog getrequestlog&limit=10
Registo de avisos	getwarnlog getwarnlog&limit=10

6.3 *Dashboard*

Nesta secção é apresentado o *interface web* desenvolvido na construção do protótipo, assim como as suas principais funcionalidades. Este *interface* permite maior interacção do administrador com o sistema durante o seu funcionamento, visualizando o seu funcionamento e se necessário, efectuar operações de administração em tempo real.

De seguida, são descritas as principais funcionalidades desta *interface*, que neste momento, apenas pode ser acedida pelos administradores (ainda não estão definidos perfis de utilizadores por se tratar de um protótipo).

Podemos visualizar todos os dispositivos de rede (ver figura 6.5), com a descrição do tipo de dispositivo, endereços 64 *bits* e 16 *bits* de rede, a última actualização, o intervalo máximo que pode estar sem comunicar, o número de sensores agregados e, por fim, o seu estado actual (*online* ou *offline*).

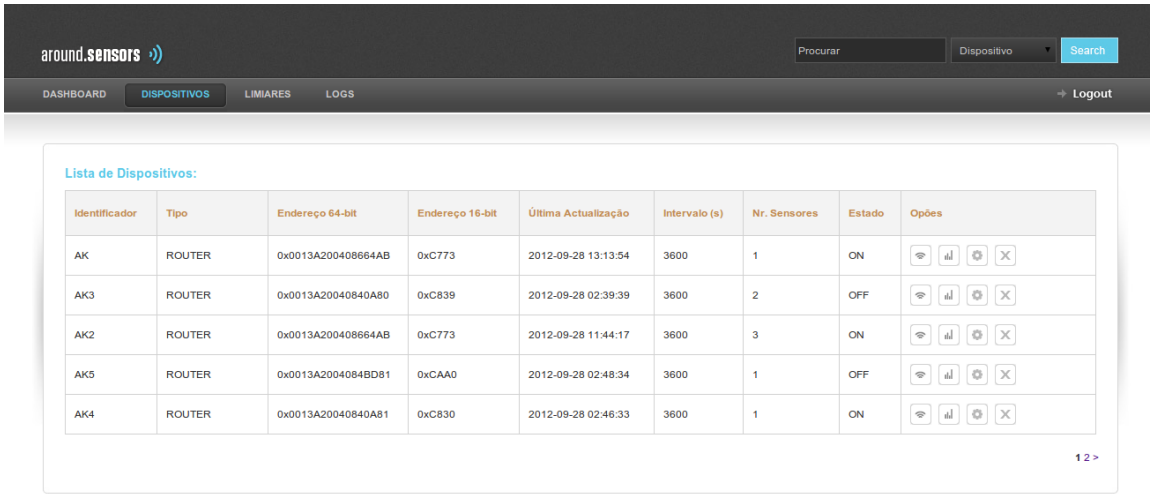

Figura 6.5: Visualização dos dispositivos

É ainda permitido remover um sensor específico (caso tenha sido removido da rede) e visualizar gráficos sobre o RSSI das mensagens entregues e a taxa de transferência de mensagens para esse sensor específico.

Outra funcionalidade é visualizar os sensores agregados a um dispositivo de rede específico (ver figura 6.6), com a última actualização, o intervalo máximo que pode estar sem comunicar, o número de sensores agregados e, por fim, o seu estado actual (*online* ou *offline*).

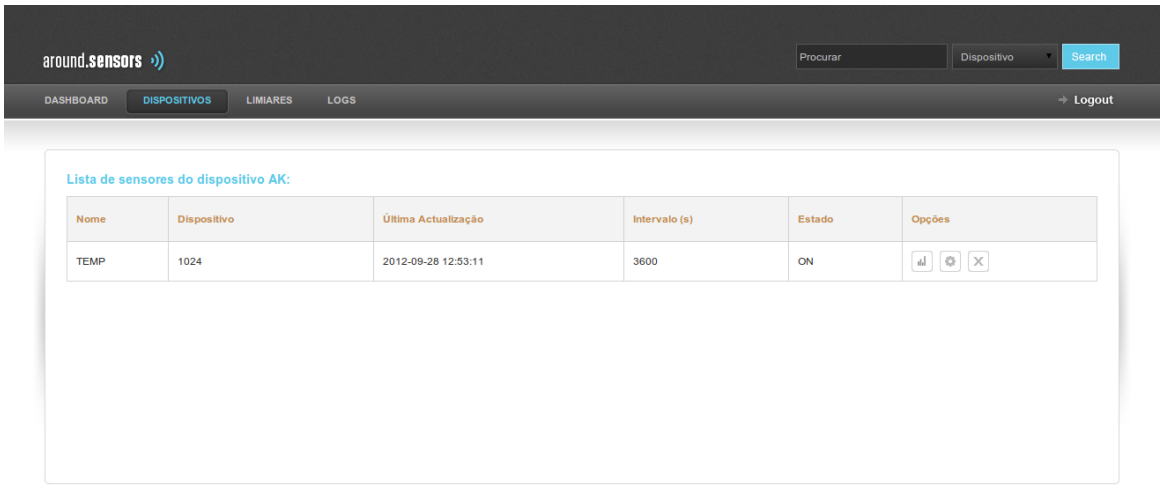

Figura 6.6: Visualização dos sensores agregados a um dispositivo específico

Novamente, é permitido remover um sensor específico (caso tenha sido removido do dispositivo de rede) e visualizar um gráfico temporal com valores recolhidos por esse sensor (ver figura 6.7). Desta forma, é possível facilmente verificar os consumos de cada sensor e detectar, por exemplo, picos de utilização nos sensores de electricidade e de gás.

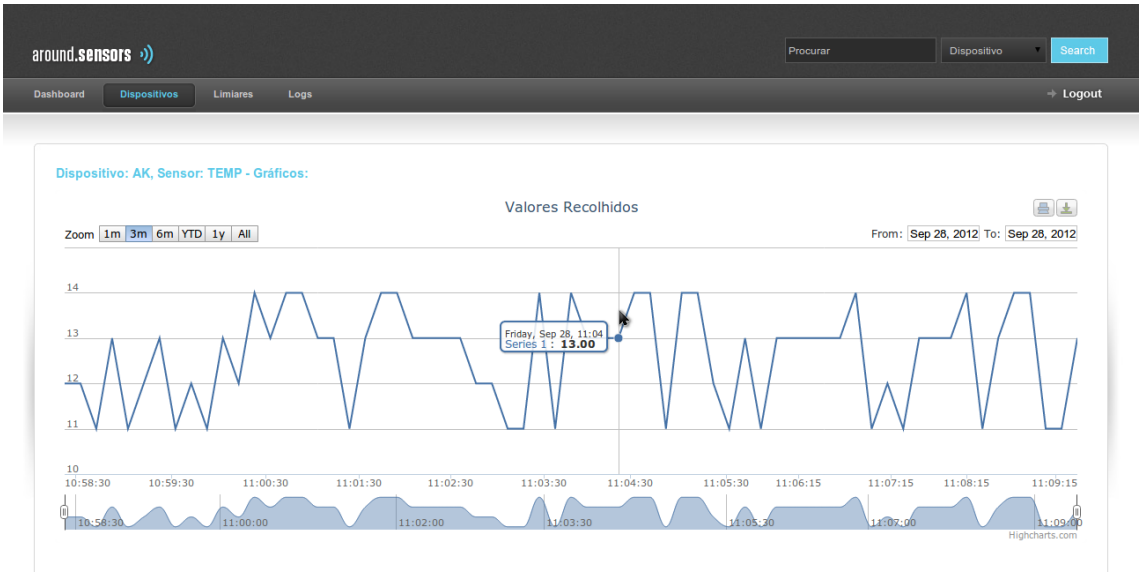

Figura 6.7: Visualização dos valores recolhidos por um sensor específico

Relativamente aos limiares que estão associados aos sensores, estes têm que ser inseridos pelo administrador para que o sistema os utilize. Por este motivo, existe uma

secção na *interface* (ver figura 6.8)que permite adicioná-los, mas também visualizar os parâmetros dos actuais, podendo mesmo modificá-los ou adicionar novos.

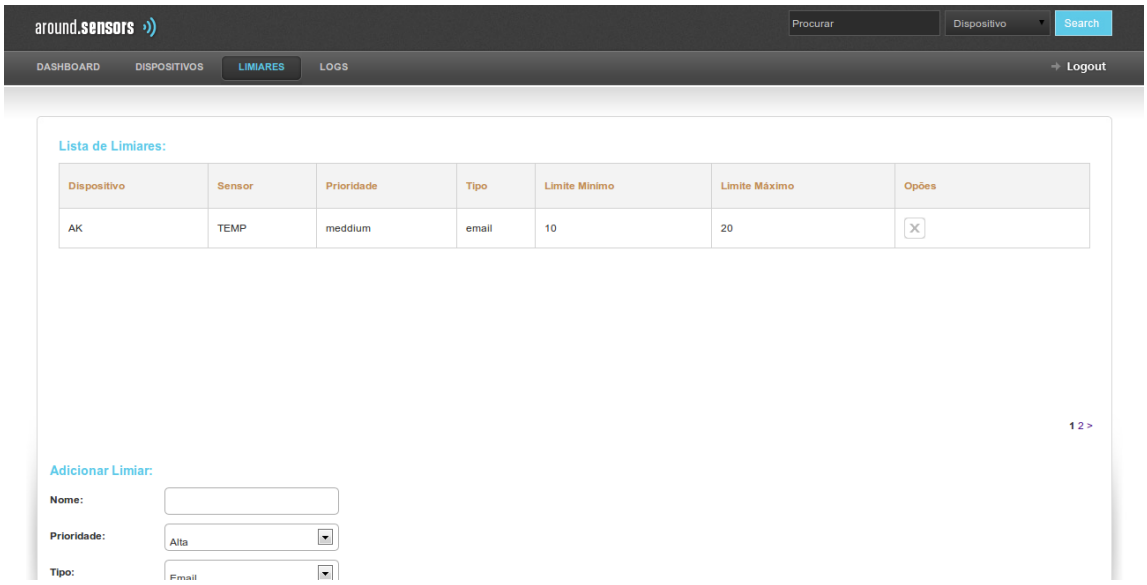

Figura 6.8: Visualização dos limiares associados aos sensores

Finalmente, é permitido visualizar os ficheiros de *log* gerados pelo sistema em tempo real (ver figura 6.9), que se encontram divididos em quatro secções: avisos, *debug*, erros e pedidos efectuados. Assim, a detecção e correcção de anomalias ocorridas no sistema tornam-se tarefas mais facilitadas.

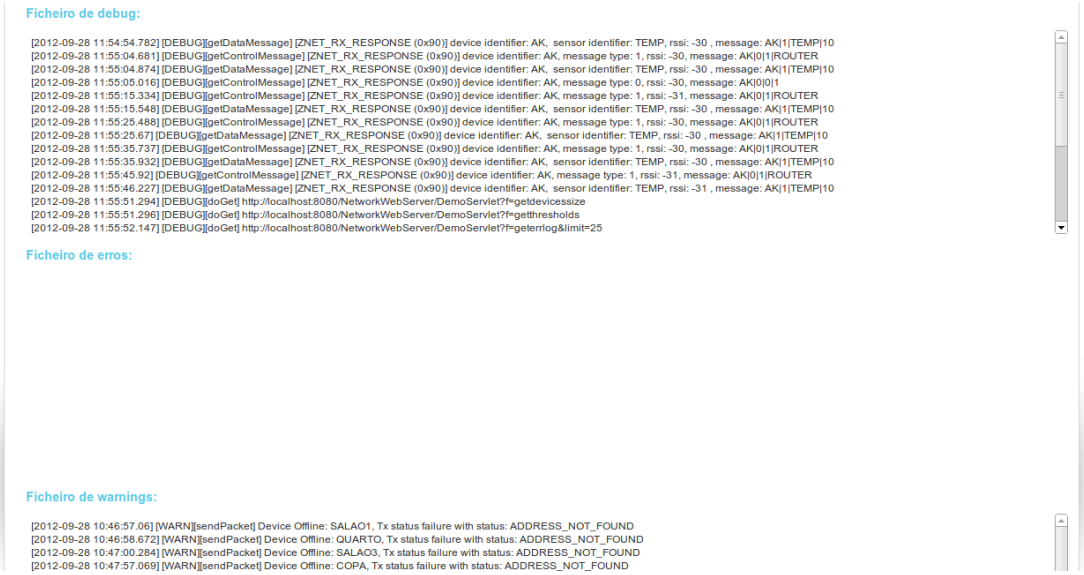

Figura 6.9: Visualização dos *logs* gerados pelo sistema

Destaque ainda para o facto de não ser necessário inserir dispositivos e sensores

no sistema, pois este tem a capacidade de detectá-los, pelo que os insere de forma automática, incluindo todos os parâmetros associados. Apenas pode ser modificado o tempo máximo em que cada dispositivo/sensor pode estar sem comunicar (por defeito ele verifica de hora a hora).

6.4 Arduino *Sketch*

Este processo contempla um série de mensagens estruturadas que são enviadas para o coordenador da rede. Antes de efectuar o *upload* de código, é necessário efectuar algumas alterações: identificar o dispositivo, o tipo de dispositivo, os sensores de recolha e o endereço do coordenador de rede. No apêndice D.2, podemos visualizar um excerto de código que mostra exactamente isso.

O Arduino é um *hardware* muito limitado, comparativamente ao *hardware* existente no DL, pelo que tudo se encontra em memória e não é feito qualquer registo do funcionamento do processo. A rede é que está responsável por gerir o seu funcionamento.

No apêndice D.1, temos os passos necessários para efectuar um *upload* de código no Arduino Fio.

Capítulo 7

Conclusões e Trabalho Futuro

Neste capítulo são apresentadas as conclusões tiradas ao longo da realização deste projecto e funcionalidades que poderão vir a ser incluídas no mesmo.

7.1 Conclusões

Este projecto visava dotar a rede eléctrica de informação e de equipamentos capazes de automatizar a gestão das redes, melhorar a qualidade de serviço, diminuir os custos de operação, promover a eficiência energética e a sustentabilidade ambiental, potenciar a utilização das energias renováveis e do veículo eléctrico.

Para além disso, permitirá controlar e gerir a rede de distribuição eléctrica, em tempo real, permitindo assim, que as empresas envolventes disponibilizem informações, produtos e vários tipos de serviços de valor acrescentado para os consumidores.

Resumidamente, este projecto surgiu com a necessidade de analisar e corrigir algumas falhas conhecidas na rede de energia criada por um fornecedor de energia português. Assim, de seguida são apresentados os meus contributos em cada objectivo descrito no Capítulo 1:

- ***Objectivo 1:*** Os testes realizados à componente rádio, as várias topologias que a rede apresenta, a inclusão de ACKs nas mensagens, o algoritmo de encaminhamento utilizado pelo ZigBee e as normas apresentadas no presente relatório ajudam a solucionar ou minimizar muitos dos problemas relativos à fiabilidade das comunicações sem fios. A alimentação constante contribui para a disponibi-

lidade dos dispositivos de rede.

- ***Objectivo 2:*** Relativamente à segurança da rede, a inclusão dos mecanismos de segurança definidos pelo ZigBee, tais como, chaves de rede e de ligação, MICs, *frame counters*, permitiram solucionar os problemas de confidencialidade e integridade dos sensores e das mensagens existentes, assim como a sua autenticidade e não repúdio.

- ***Objectivo 3:*** De um modo geral, a monitorização da rede foi conseguida. É possível controlar os dispositivos de rede e correspondentes sensores através do DL, com tempos de sincronização e limiares de dados, ou ainda através de pedidos remotos aos dispositivos. No entanto, a inclusão de novas regras ao longo do tempo podem contribuir para o seu melhoramento e eficácia. A sua incorporação pode surgir com a necessidade e o crescimento da rede, aumentando a sua maturidade e autonomia.

- ***Objectivos 4 e 5:*** O envio de *emails* automático por parte do sistema aquando da detecção de uma anomalia, o registo do seu funcionamento em ficheiros *log* e a visualização e alteração de parâmetros de rede remotamente sustentam ainda mais a independência da rede.

Para além dos objectivos iniciais definidos, foi construída uma *interface web* simples, que visa utilizar todos os métodos disponibilizados pelo sistema. Assim, operações como instalar, alterar ou remover dispositivos e sensores presentes na rede, bem como visualizar as informações recolhidas sobre a rede ou pelos sensores, tornam-se mais simples e intuitivas para o instalador e o administrador da rede.

Em relação ao *software* utilizado para configurar a rede (X-CTU), e apesar de ser possível configurar os módulos XBee através do terminal (CoolTerm, HyperTerminal, Tera Term, entre outros) [7], este é destacadamente o que mais ajuda o utilizador na configuração.

A utilização das APIs Arduino-XBee e XBee-API tornaram-se numa mais valia para a realização do projecto, pois permitiram não só a ligação aos módulos de rádio, como a utilização de métodos já implementados relativos à recepção, controlo e envio de mensagens na rede.

De realçar novamente, foram identificadas algumas alternativas ao *hardware* utilizado no DL e constatado que existem soluções mais viáveis que a actual, pelo que, assim que

possível, é recomendável a sua substituição. Este é também um mercado emergente, por isso é natural que brevemente surjam novas soluções atraentes.

7.2 Trabalho Futuro

Este projecto num futuro próximo pode ser complementado com a inserção de novas funcionalidades, nomeadamente ao nível de controlo e gestão da rede. A rede pode ser expandida a servidores centrais que contêm todas as informações das sub-redes. Cada coordenador da rede tem associado a si o processo de recolha de informações desenvolvido e os servidores centrais utilizam pedidos HTTPS para obter respostas de uma sub-rede específica.

O *dashboard* pode ser melhorado, não só a nível gráfico como a nível de funcionalidades. Portanto, sempre que seja incluída uma nova funcionalidade, este deve ser actualizado, permitindo que o administrador/cliente possa usufruir dela, desde que possua as permissões de segurança necessárias.

Outra possibilidade é alargar o protótipo criado a outras áreas, nomeadamente a aplicações de domótica, modificando os sensores de recolha ou acrescentando actuadores. Neste caso em concreto, a empresa Around Knowledge está a estudar a possibilidade de vir a utilizá-lo juntamente com lâmpadas inteligentes.

Um mercado emergente na actualidade é o das aplicações móveis. A criação de aplicações móveis para os sistemas operativos mais comuns (iOS e Android) pode torna-se uma mais valia para o projecto. Neste aspecto, poderíamos ter dois tipos de aplicações: para os técnicos e para os clientes. As aplicações para os técnicos permitiriam, por exemplo, que este chegasse a um estabelecimento e pudesse configurar a rede e/ou adicionar novos dispositivos através do auxílio de uma aplicação no seu *smartphone*. Outro possível exemplo, seria existir aplicações que continham informações técnicas do funcionamento e gestão da rede, permitindo a recepção de notificações sempre que ocorressem anomalias. Relativamente às aplicações para os clientes, permitiriam controlar os seus consumos e gastos monetários da sua energia.

Se este projecto for alargado à área da domótica, estes podem mesmo utilizar os actuadores nas suas casas para ligar/desligar lâmpadas, ar condicionados e outros electrodomésticos através do seu *smartphone*.

Apêndice A

Módulos XBee

A.1 Ligação

1. Ligar o XBee Explorer USB com um módulo XBee ao computador e executar o *software* X-CTU;
2. Ligar-se ao módulo na secção *"PC Settings"* do *software* através da porta de série com as especificações:
    - *Baud Rate*: 9600
    - *Flow Control*: *None*
    - *Data Bits*: 8
    - *Parity*: *None*
    - *Stop Bits*: 1

A.2 Configurar Dispositivo ZC (Modo API)

1. Escolher o *modem "XBP24-ZB"* e o *firmware "ZigBee Coordinator API"* na secção *"Modem Configuration"*;
2. Escolher a rede a usar (PAN ID);
3. Configurar o endereço de destino com o endereço do módulo de destino (endereço visível na parte de trás);

4. Identificá-lo como COORDENADOR (NI)[1];
5. Activar o modo API 2;
6. Guardar as configurações no módulo;

A.3 Configurar Dispositivo ZR (Modo API)

1. Escolher o *modem "XBP24-ZB"* e o *firmware "ZigBee Router AT"* na secção *"Modem Configuration"*;
2. Escolher a rede a usar (com PAN ID do COORDINATOR);
3. Configurar o endereço de destino com o endereço do módulo de destino (endereço visível na parte de trás);
4. Identificá-lo como ROUTER (NI)[1];
5. Activar o modo API 2;
6. Guardar as configurações no módulo;

A.4 Configurar Dispositivo ZED (Modo API)

1. Escolher o *modem "XBP24-ZB"* e o *firmware "ZigBee End Device AT"* na secção *"Modem Configuration"*;
2. Escolher a rede a usar (com PAN ID do COORDINATOR);
3. Configurar o endereço de destino com o endereço do coordenador (endereço visível na parte de trás);
4. Identificá-lo como ENDDEVICE (NI)[1];
5. Activar o modo API 2;
6. Guardar as configurações no módulo;

[1]Para compreender melhor o funcionamento da rede, é aconselhável que o parâmetro NI dos dispositivos de rede identifique-o facilmente. Como exemplo, podemos identificá-los com a posição física (p.e. SALAO)

A.5 Aplicar Segurança

A.5.1 Pré-Configuração das Chaves de Ligação

As configurações para o ZC são:

1. *ID:* 123 (arbitrariamente seleccionado)
2. *EE:* 1
3. *NK:* 0
4. *KY:* 1234 (arbitrariamente seleccionado, mas diferente de 0)
5. Gravar parâmetros no módulo.

As configurações para o ZR/ZED são:

1. *ID:* 123
2. *EE:* 1
3. *KY:* 1234
4. Gravar parâmetros no módulo.

A.5.2 Obtenção das Chaves durante a Ligação à Rede

As configurações para o ZC são:

1. *ID:* 123 (arbitrariamente seleccionado)
2. *EE:* 1
3. *NK:* 0
4. *KY:* 0
5. Gravar parâmetros no módulo.

As configurações para o ZR/ZED são:

1. *ID:* 123
2. *EE:* 1
3. *KY:* 0
4. Gravar parâmetros no módulo.

Apêndice B

Comandos AT

ID	Descrição
DH	***Destination Address High***. Verificar/colocar os 32 *bits* mais altos do endereço de destino de 64 *bits*. Quando combinado com o DL, define o endereço de destino de 64 *bits* para a transmissão de dados. Definições especiais para DH e DL incluem 0x000000000000FFFF (*endereço broadcast*) e 0x0000000000000000 (endereço do coordenador).
DL	***Destination Address Low***. Verificar/colocar os os 32 *bits* mais baixos do endereço de destino de 64 *bits*. Quando combinado com o DH, define o endereço de destino de 64 *bits* para a transmissão de dados. Definições especiais para DH e DL incluem 0x000000000000FFFF (*endereço broadcast*) e 0x0000000000000000 (endereço do coordenador).
SH	***Serial Number High***. São os 32 *bits* mais altos do endereço de 64 *bits* do módulo.
SL	***Serial Number Low***. São os 32 *bits* mais baixos do endereço de 64 *bits* do módulo.
MY	***16-bit Network Address***. Endereço de rede de 16 *bits* do módulo. O valor 0xFFFE significa que o módulo não faz parte de uma rede ZigBee.
NI	***Node Identifier***. *String* ASCII que identifica o módulo.
ID	***Extended PAN ID***. Identificador da PAN ID (32 *bits*). Caso seja o valor 0, o coordenador seleciona um ID aleatório.

EE	***Encryption Enable***. Verificar/iniciar o modo de encriptação.
FV	***Firmware Version***. Versão do *firmware* do módulo.
HV	***Hardware Version***. Versão do *hardware* do módulo. Distingue diferentes plataformas.
DB	***RSSI Last Packet***. Valor do RSSI do último pacote recebido.

Apêndice C

Excertos de Código

Listing C.1: Detecção do estado do pacote transmitido

```
ZNetTxStatusResponse resp = (ZNetTxStatusResponse) response;

if (resp.getDeliveryStatus() != DeliveryStatus.SUCCESS) {
  errors++;

  DeliveryStatus status = resp.getDeliveryStatus();
  switch (status) {
  case NETWORK_ACK_FAILURE:
    ackErrors++;
    break;
  case CCA_FAILURE:
    ccaErrors++;
    break;
  case ADDRESS_NOT_FOUND:
    purgeErrors++;
    break;
  case PAYLOAD_TOO_LARGE:
    maxPayloadSizeErrors++;
    break;
  default:
    System.err.println("Tx status: " + status);
    break;
  }

  log.error("Tx status failure with status: "
      + resp.getDeliveryStatus());
  return FAILURE;
```

```
} else {
  return DELIVERED;
}
```

Detecção do estado do pacote enviado pelo transmissor. Assim, é possível saber se o pacote foi recebido pelo destino e em caso de falha, o motivo que causou a essa falha.

Listing C.2: Envio de um pacote

```
ZNetTxRequest request = new ZNetTxRequest(xbeeAddr64, payload);

log.debug("Sending request to XBee: " + request);

long start = System.currentTimeMillis();

ZNetTxStatusResponse response = (ZNetTxStatusResponse) xbee
    .sendSynchronous(request, timeout);
countPackets++;

long responseTime = System.currentTimeMillis() - start;
log.info("Received response in [" + responseTime + "]: " + response);
sumResponseTime += responseTime;
```

Envio de um pacote por parte do transmissor. É obtido o tempo que demora a chegar a correspondente confirmação (RTT). Se a confirmação não chegar até que o *timeout* (p.e. 10000 ms) exceda, assume-se que o pacote não foi entregue.

Apêndice D

Arduino

D.1 Instalação de um *sketch* no Arduino Fio

Existem 2 formas de fazer *upload* de novos *sketchs* no Arduino Fio: por cabo, usado o cabo *FTDI USB-to-serial* ou o adaptador *USB-to-serial*, ou sem fios, através de um par de módulos XBee. Apenas foi usado a primeira forma, pois como foram usados módulos XBee série 2, estes não permitem o *upload* de *sketchs* sem fio [26].

Depois de estabelecer correctamente a ligação entre o Arduino Fio e um computador (ver em [26]) e instalado o *software* Arduino IDE (em [25]), vamos ao menu *Tools→Board* e escolhemos *"Arduino Fio"*. De seguida é só escolher a porta série à qual temos o Arduino Fio ligado e está tudo operacional para efectuar *upload* de *sketchs*.

Durante o *upload*[1], o Arduino Fio não nos indica se o *upload* foi feito com sucesso, apenas é possível ver os *leds* TX e RX a piscar. No entanto, o *software* dá-nos esta informação.

D.2 Excerto do *Sketch*

Listing D.1: Excerto do *Sketch*

```
// Identificacao
char deviceIdentifier[] = "SALAO";
char deviceType[] = "ROUTER";
```

[1]Não devem existir módulos XBee ligados ao Arduino Fio

```
char temperatureSensor[] = "TEMPERATURA";

(...)

// Endereco do coordenador de rede
XBeeAddress64 addr64 = XBeeAddress64(0x0013a200, 0x4086a3a0);

(...)

// Mensagens de controlo
zbTx_keep = ZBTxRequest(addr64, keep_alive, sizeof(keep_alive));
zbTx_type = ZBTxRequest(addr64, device_type, sizeof(device_type));

// Mensagens de dados
zbTx_data = ZBTxRequest(addr64, payload_data, sizeof(payload_data));

(...)

// Envio de um pacote de dados
xbee.send(zbTx_data);
```

Referências

[1] RX Packet (16-bit address) Frames. http://3.bp.blogspot.com/-YjBkRh9dBc4/ThAa2oFmm5I/AAAAAAAAAQg/HRtWD6IYe5Q/s1600/xbee_api_rx_frame.png, Agosto 2012.

[2] TinyOS. http://www.tinyos.net/, Agosto 2012.

[3] Waspmote vs squidbee. http://www.libelium.com/squidbee/index.php?title=Waspmote_vs_SquidBee, Agosto 2012.

[4] ZigBee Protocol Stack. http://infotechplus.free.fr/image/stack.png, Agosto 2012.

[5] Digi. X-CTU Configuration & Test Utility Software. Technical report, Digi International Inc, 2008.

[6] Digi. XBee/XBee-PRO ZB RF Modules. Technical report, Digi International Inc, 2010.

[7] R. Faludi. *Building Wireless Sensor Networks: with ZigBee, XBee, Arduino, and Processing*. Oreilly Series. O'Reilly Media, Incorporated, 2010.

[8] S. Farahani. *ZigBee Wireless Networks and Transceivers*. Newnes/Elsevier, 2008.

[9] The Raspberry Pi Foundation. Raspberry Pi. http://www.raspberrypi.org/, Consultado em Maio 2012.

[10] Wireshark Foundation. wireshark Home Page. http://www.wireshark.org/, Consultado em Setembro 2012.

[11] D. Gislason. *ZigBee Wireless Networking*. IT Pro. Elsevier, 2008.

[12] Inc. GLOBALSCALE Technologies. Dreamplug. http://www.globalscaletechnologies.com/p-54-dreamplug-devkit.aspx, Consultado em Janeiro 2012.

[13] Inc. GLOBALSCALE Technologies. Dreamplug. `http://code.google.com/p/dreamplug/`, Consultado em Janeiro 2012.

[14] Digi International Inc. XBee Wireless RF Modules. `http://www.digi.com/xbee/`, Consultado em Fevereiro 2012.

[15] MEMSIC Inc. IRIS - Wireless Measurement System. Technical report, MEMSIC Inc.

[16] H. Karl and A. Willig. *Protocols and Architectures for Wireless Sensor Networks*. John Wiley & Sons, 2007.

[17] Around Knowledge. Around Knowledge. `http://www.aroundknowledge.com/`, Consultado em Maio 2012.

[18] Daintree Networks. Building and Operating Robust and Reliable ZigBee Networks. Technical report, Daintree Networks Inc, Agosto 2008.

[19] Daintree Networks. ZigBee Security. `http://www.daintree.net/resources/security.php`, Consultado em Maio 2012.

[20] Daintree Networks. Getting Started with ZigBee and IEEE 802.15.4. Technical report, Daintree Networks Inc, Fevereiro 2008.

[21] J. Oxer and H. Blemings. *Practical Arduino: Cool Projects for Open Source Hardware*. Technology in Action. Apress, 2009.

[22] A. Rapp. A Java API for Digi XBee/XBee-Pro OEM RF Modules. `http://code.google.com/p/xbee-api/`, Consultado em Março 2012.

[23] A. Rapp. Arduino Library for Communicating with XBees in API Mode. `http://code.google.com/p/xbee-arduino/`, Consultado em Março 2012.

[24] Libelium Comunicaciones Distribuidas S.L. SquidBee Main Page. `http://www.libelium.com/squidbee/index.php?title=Main_Page`, Consultado em Fevereiro 2012.

[25] Arduino Team. Arduino Development Environment. `http://arduino.cc/en/Guide/Environment`, Consultado em Março 2012.

[26] Arduino Team. Arduino Fio Programming. `http://arduino.cc/en/Main/ArduinoBoardFioProgramming`, Consultado em Março 2012.

[27] Arduino Team. Arduino Main Page. `http://www.arduino.cc/`, Consultado em Março 2012.

[28] Arduino Team. Official Arduino Boards. `http://arduino.cc/en/Main/Hardware`, Consultado em Março 2012.

[29] VIA Technologies. APC Main Page. `http://apc.io/`, Consultado em Junho 2012.

[30] T. Zia and A. Zomaya. Security issues in wireless sensor networks. In *Systems and Networks Communications, 2006. ICSNC '06. International Conference on*, page 40, oct. 2006.