Luís Pedro Marques Brás

DESENVOLVIMENTO DE SISTEMA DE LOCALIZAÇÃO INDOOR DE BAIXO CONSUMO

Luís Pedro Marques Brás

DESENVOLVIMENTO DE SISTEMA DE LOCALIZAÇÃO INDOOR DE BAIXO CONSUMO

Luís Pedro Marques Brás

DESENVOLVIMENTO DE SISTEMA DE LOCALIZAÇÃO INDOOR DE BAIXO CONSUMO

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Electrónica e Telecomunicações (Mestrado Integrado), realizada sob a orientação científica do Dr. Nuno Borges de Carvalho, Professor Associado do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro, e co-orientação do Dr. Pedro Pinho, Professor Adjunto do Instituto Superior de Engenharia de Lisboa.

O júri

Presidente

Prof. Dr. José Carlos Esteves Duarte Pedro

Professor Catedrático da Universidade de Aveiro

Vogais

Prof. Dr. Rafael Ferreira da Silva Caldeirinha

Professor Coordenador da Escola Superior de Tecnologia e Gestão de Leiria, Politécnico de Leiria

Prof. Dr. Nuno Miguel Gonçalves Borges de Carvalho

Professor Associado da Universidade de Aveiro (Orientador)

Prof. Dr. Pedro Renato Tavares de Pinho

Professor Adjunto do Instituto Superior de Engenharia de Lisboa (Co-orientador)

"The wireless telegraph is not difficult to understand. The ordinary telegraph is like a very long cat. You pull the tail in New York, and it meows in Los Angeles. The wireless is the same, only without the cat "

Albert Einstein

Agradecimentos

Nesta dissertação teria obrigatoriamente que referir a minha imensa gratidão a várias pessoas que directa ou indirectamente me ajudaram na execução deste trabalho.

À minha família e amigos próximos que me apoiaram e incentivaram ao longo de todo o meu percurso académico.

Aos meus colegas do laboratório de rádio-frequência que permitiram um ambiente de discussão muito enriquecedor, essencial para um bom local de trabalho.

Ao Engenheiro Arnaldo Monteiro que sempre se mostrou disponível para me esclarecer várias dúvidas que tive ao longo da execução da dissertação.

Ao Professor Rocha Pereira e ao meu co-orientador Pedro Pinho pelos conhecimentos que me transmitiram sobre antenas.

Um agradecimento especial ao meu orientador Nuno Borges de Carvalho que além da orientação na elaboração da dissertação me proporcionou experiências muito enriquecedoras a nível curricular.

Palavras-Chave

Indicação de Intensidade do Sinal Recebido (RSSI), Localização Indoor, fingerprinting, Redes Neuronais, interface com o utilizador, SoC cc2430/31, sinal de Rádio Frequência (RF), ZigBee

Resumo

Actualmente, os sistemas de localização apresentam-se como uma área em forte expansão. Os sistemas de localização possibilitam inúmeras vantagens para o controlo e gestão a nível empresarial como a nível residencial. Sendo o consumo energético um factor de tremenda importância destes.

Esta dissertação propõe um sistema de localização *indoor* de baixo consumo energético baseado na plataforma *CC2431ZDK* utilizando a tecnologia *ZigBee*. Foi também desenvolvida uma interface gráfica em linguagem *Java* de fácil utilização onde é possível ao utilizador configurar e seleccionar o modo de localização desejado. Esta poderá ser baseada em proximidade, maior nível de potência detectado por sensores de referência ou numa localização de maior exactidão através de uma rede neuronal. Por fim, foi proposto o uso de uma antena com diagrama de radiação mais adequado para uma localização por proximidade comparativamente às antenas fornecidas na plataforma *CC2431ZDK*, deste modo foi desenhada e analisada uma antena impressa.

Abstract

Nowadays, the location systems appears as an area in strong expansion. The location systems enable advantage for the control and management at the enterprise residential level. One of the most important factors in their implementation is the energy consumption.

This dissertation proposes a system for indoor location of low power consumption based on *CC2431ZDK* platform using *ZigBee* technology. It was also developed a graphical interface in *Java* language, easy to use where the user can configure and select the desired mode of detection. This can be based in proximity, higher level of power detected by reference nodes or a location of higher precision through a neural network. Finally, was proposed the use of an antenna with more appropriate radiation diagram for a location by proximity, comparatively to the antennas provided by the *CC2431ZDK* platform, thus was designed and analyzed a *microstrip* antenna.

Índice de conteúdos

Índice de ilustrações

Índice de tabelas

Acrónimos:

AF - *Application Framework*

AOA - *angle of arrival*

APDU - *Application Protocol Data Unit*

APS - *Application Support SubLayer*

BPSK - B*inary Phase-Shift Keying*

CAP- *contention access period*

CCA - *clear channel assessment*

CFP - *contention free period*

CSMA-CA - *Carrier Sense Multiple Access – Collision Avoidance*

EB – *Evaluatin Board*

ED – *Energy Detection*

FANN - *Fast Artificial Neural Network Library*

FFD – *Full Function Device*

GLONASS - *Global Orbiting Navigation Satellite System*

GPS - *Global Posotioning System*

GTS - *guaranteed time slot*

Hi RSSI - *High Received Signal Strengh Indication*

ISM – *Industrial, Scientific and Medical*

JVM - *Java Virtual Machine*

kNN - *k-nearest neighbor*

LANDMARC - *Location Identification based on Dynamic Active RFID Calibration*

LOS - *Line of Sight*

LQI – *Link Quality Indication*

MAC – *media access control*

MFR - *MAC footer*

MLP – *multi-layer-perceptron*

MPDU - *MAC Protocol Data Unit*

MSDU - *MAC Service Data Unit*

NLDE – *Network Layer Data Entity*

NLOS - *Non-line-of-sight ou Near-line-of-sight*

NPDU - *Network Protocol Data Unit*

O-QPSK - *Offset – Quadrature Phase-Shift Keying*

OSI - *Open Systems Interconnections*

PAN - *Personal access Network*

PD – *PHY Data*

PLME - *Physical Layer Management Entity*

PPDU - *PHY protocol data unit*

PSDU - *PHY Service Data Unit*

RF – *radio-frequency*

RFD – *Reduced Function Device*

RFID - *Radio-Frequency Identification*

RSSI - *Received Signal Strengh Indication*

RTLS -*real time locating systems*

RTOF - *Roundtrip time of flight*

SAP - *Service Access Point*

SFD - *Start of Frame Delimiter*

SHR - *Synchronization Header*

SOC – *System-On-Chip*

SOC_BB - *System on Chip battery board*

TCP/IP – *Transmission Control Protocol/Internet Protocol*

TDA - *time difference of arrival*

TI - *Texas Instruments*

TOA – *time of arrival*

TOF - *time of flight*

UWB - *Ultra-Wide Band (EEE 802.15.3)*

Wi-Fi - *Wireless Fidelity (IEEE 802.11)*

ZDO - *Zigbee Device Object*

Capítulo 1

Introdução

1.1 . Descrição geral

A localização de pessoas ou objectos é hoje em dia um tema de pesquisa de bastante interessa tanto a nível académico como a nível industrial.

Várias aplicações hoje em dia necessitam ou fazem uso da informação de localização de pessoas ou objectos tais como: segurança automóvel, navegação área ou marítima, controlo de *stocks* de mercadorias, controlo estatístico de preferência dos clientes, activação de eventos através da proximidade de pessoas, entre outros.

Os sistemas de localização dizem respeito a um conjunto de dispositivos, técnicas, algoritmos e aplicações que em conjunto estimam as coordenadas absolutas ou relativas de uma pessoa ou objecto num determinado ambiente de localização.

A função inicial que deu origem aos sistemas de localização foi o suporte a aplicações militares em ambientes externos (*outdoor*). A infra-estrutura deste tipo de sistemas é usualmente formada por estações terrestres e uma constelação de satélites, possibilitando assim a obtenção geográfica da localização. O sistema norte-americano *GPS (Global*

Posotioning System) [1] e o sistema russo *GLONASS (Global Orbiting Navigation Satellite System)* [2] são exemplos deste tipo de sistemas.

Hoje em dia a infra-estrutura da rede celular também possibilita serviços de localização em ambientes *outdoor* através de técnicas como o *Cell ID, O-TDOA (Oriented Time Difference if Arrival)* e o *A-GPS (Assisted GPS)* [3].

Os primeiros sistemas de localização interiores (*indoor)* foram baseados em sinais infravermelhos e em "tapetes" especiais que incorporavam sensores de pressão [4].

Nos sistemas baseados em infravermelhos, receptores captam os sinais de identificação emitidos por transmissores infravermelhos acoplados a pessoas ou objectos que se pretendem localizar. No caso de sistemas baseados em sensores de pressão a informação é obtida com a activação dos sensores através do peso da pessoa ou objecto.

Nos últimos anos, o desenvolvimento de dispositivos móveis e redes sem fios de curto alcance levaram a um desenvolvimento de sistemas de localização *indoor* baseados em sinais de rádio-frequência (*RF*).

Estes sistemas de localização usam diversas tecnologias de comunicação, entre elas: *RFID (Radio-Frequency Identification), Wi-Fi (Wireless Fidelity – IEEE 802.11), Bluetooth (IEEE 802.15.1), UWB (Ultra-Wide Band – IEEE 802.15.3) e ZigBee (IEEE 802.15.4).*

A tecnologia *ZigBee [5]* revelou-se bastante promissora devido a várias características tais como: baixo consumo, pilha protocolar simples, admissão de várias topologias de rede, baixa latência, entre outras, descritas com maior pormenor no capítulo *ZigBee*.

A estimativa da localização através desta tecnologia é baseada na potência do sinal *RF*. Esta pode ser realizada pelos próprios terminais da rede (*end devices)* ou através de sinais captados pelos sensores de referência (*routers*) que posteriormente reencaminham as informações para o servidor que estima a localização de acordo com um determinado algoritmo de localização. O primeiro modo representa o funcionamento típico do *CC2431 Location Engine* [6] da *Texas Instruments (TI)* e o segundo representa o modo adoptado para o sistema desenvolvido nesta dissertação, onde pretendemos o mínimo consumo energético por parte dos *end devices*.

A estimativa da localização baseada neste tipo de sinal depende de diversos factores inerentes à natureza do próprio sinal. Estes sinais em ambientes *indoor* não são estáveis, onde podem sofrer reflexão, absorção, espalhamento e difracção do sinal. A presença de

obstáculos (mesas, cadeiras, armários), o número e disposição das pessoas, a própria geometria do ambiente influenciam a estabilidade do sinal de rádio-frequência podendo por vezes levar a perdas significativas da potência do sinal [9].

Esta variabilidade da potência do sinal *RF* faz com que a posição estimada da localização das pessoas ou objectos se distancie da posição real destes.

Para reduzir este erro usualmente são utilizados algoritmos de localização que possibilitam um suporte ao processo de localização. Os algoritmos de localização baseados em assinaturas *RF* (f*ingerprinting)* mais utilizados até hoje são baseados em métodos probabilísticos; detecção por vizinhança mais próxima *(k-nearest neighbor – kNN)* e redes neuronais [7].

Esta dissertação propõe um sistema de localização baseado na tecnologia *ZigBee* com o suporte da plataforma *CC2431ZDK da Texas Instruments.*

O sistema proposto permite um baixo consumo energético por parte dos dispositivos terminais (*end devices)*, configuração dos sensores usados para o processo de localização, escolha entre macro-localização e localização mais fina.

O objectivo principal do sistema proposto foi o mínimo consumo energético, deste modo foi alterada a programação dos *end devices*, onde após o registo com a rede, ciclicamente adormecem durante um tempo determinado e acordam enviando *N* sinais *RF* para a rede. O facto de não haver nenhum tipo de troca de mensagens além do registo com a rede leva a uma tremenda redução energética por parte destes módulos.

A escolha dos roteadores (*routers)* que irão ser considerados para a localização permite uma maior eficiência temporal do sistema assim como uma rápida e fácil alteração da rota do sistema em caso de alteração do ambiente de localização.

A macro-localização permite a detecção de todos os sensores baseada em proximidade, maior nível de potência detectada, onde será atribuída a localização dos *end devices* ao *router* que detectar o nível de potência mais elevado, designado por *Hi RSSI (Hi Received Signal Strengh Indication)*.

O outro modo de localização permite a localização de um *end device* escolhido, sendo esta uma localização mais fina baseada em redes neuronais. Este tipo de localização implica uma calibração do sistema na zona de detecção (fase *offline*) que será comparada no processo de localização *online*.

A interface com o utilizador foi implementada em linguagem Java, onde a comunicação com a rede de sensores é efectuada pela ferramenta *Serial Forwarder*. Esta ferramenta possibilita a escuta de tramas TinyOS numa porta série e a criação de um servidor TCP/IP associada a esta. Isto permite a comunicação de várias aplicações, com a rede de sensores por intermediário da aplicação *Serial Forwarder,* alterada de modo a poder escutar mensagens *ZigBee*.

. Uma das grandes vantagens da linguagem *Java* está relaciona com a sua portabilidade, devido a ser uma linguagem interpretada por uma máquina virtual (*Java Virtual Machine – JVM*). Isto possibilita que aplicações *Java* possam ser executadas em qualquer plataforma ou equipamento que possua uma *JVM*.

O sistema foi analisado num ambiente com vários obstáculo e interferência electromagnética de modo a simular um ambiente *indoor* realista. As próprias antenas fornecida pelo *kit* de desenvolvimento (dipolo de meio comprimento de onda) revelaram-se pouco apropriadas para o sistema implementado, deste modo, foi proposto uma antena que melhor se adequasse ao nosso sistema. A escolha residiu numa antena *microstrip* pois apresenta um diagrama de radiação mais adequado comparativamente com às antenas fornecidas no *kit*.

1.2 . Enquadramento do projecto

Este projecto teve como base o sistema de localização LOPES [8] – LOcalização de PESsoas. Este sistema resultou de uma parceria entre o Instituto de Telecomunicações de Aveiro e três empresas: *MicroI/O*, empresa de sistemas de gestão; *MaisSis, empresa de desenvolvimento de software; e HFA, fabricante de cartões electrónicos*.

Este sistema, também baseado na plataforma *CC2431ZDK,* permite localizar pessoas em ambientes *indoor* com uma precisão inferior a 2 metros, através de 8 nós de referência instalados no tecto e com o uso de redes neuronais. Através de computadores interligados na rede é possível despoletar uma aplicação multimédia conforme a posição estimada no processo de localização. Mas apresentava certas limitações, entre elas:

- Consumo energético elevado (sensores móveis em escuta permanentemente);
- Localização apenas através de redes neuronais;
- Identificação de apenas *end devices* pré definidos na aplicação *Java*;
- Não existência de controlo do fluxo de mensagens por parte do utilizador;

- Não identificação de *routers* com problemas;
- Elevado número de mensagens enviadas na rede para o processo de localização;
- Falha da localização em caso de falha de comunicação de algum *router*.

Esta dissertação permitiu a resolução de grande parte destes problemas de modo a criar um sistema de maior eficiência energética e mais fácil utilização.

1.3 . Objectivos da dissertação

Esta dissertação tem como objectivo melhorar o sistema implementado anteriormente (LOPES) a nível do consumo energético, aplicação com o utilizador e melhoria da exactidão do sistema através da alteração da antena dos *routers.* Deste modo foi realizado:

- Uma proposta de um sistema de localização com reduzido consumo energético;
- Elaboração de uma aplicação em *Java* de fácil utilização para o utilizador;
- Aplicação de selecção dos sensores para o processo de macro-localização;
- Opção de identificação de *routers* não funcionais na aplicação;
- Opção entre escolha de macro-localização e localização mais fina baseada em redes neuronais;
- Resolução do problema de falha de mensagem, essencial para a localização baseada em rede neuronal;
- Proposta de um antena que melhor se adequasse às necessidades do sistema.

1.4 . Organização

Esta dissertação encontra-se organizada por capítulos de acordo com a seguinte descrição:

No capítulo 2 é feita uma análise das características, classificação e técnicas de localização de maior relevo consideradas até hoje. São também referidos vários algoritmos de localização mas apenas descrito o algoritmo de redes neuronais utilizado no sistema implementado.

No capítulo 3 são apresentados diferentes sistemas de localização desenvolvidos até hoje que utilizam diferentes técnicas e tecnologias, assim como uma breve descrição do seu funcionamento.

No capítulo 4 é apresentada e descrita a arquitectura protocolar *ZigBee.* São descritas

as principais características desta tecnologia assim como uma descrição detalhada das camadas em que esta se baseia.

No capítulo 5 é apresentado uma introdução a antenas *microstrip*. *O* modo como podem ser projectadas e as simulações efectuadas no desenvolvimento destas.

No capítulo 6 são descritas as plataformas de *hardware* e *software* que serviram de suporte para o desenvolvimento do trabalho realizado para esta dissertação.

No capítulo 7 é descrito o protocolo implementado na rede de sensores assim como as características e funcionalidades principais deste.

No capítulo 8 é apresentado uma descrição da interface gráfica desenvolvida assim como o seu modo de funcionamento.

No capítulo 9 são descritos os resultados dos consumos energéticos obtidos e o modo como foram medidos. Em seguida é descrito a exactidão da localização obtida segundo os dois modo de localização.

O capítulo 10 conclui esta dissertação e apresenta alguns pontos possíveis a desenvolver num trabalho futuro.

Capítulo 2

Taxonomia de sistemas de localização

Neste capítulo são apresentados os conceitos básicos de sistemas de localização. De um modo mais detalhado, na secção 2.1 . é apresentada uma proposta de taxonomia para sistemas de localização. Na secção 2.2 . são apresentadas as características dos sistemas de localização, entre elas: ambiente e modo de operação, entidade responsável pela estimativa da localização, sistemas de coordenadas utilizado, modos de representação da localização, granularidade e linha de vista entre o transmissor e receptor. A secção 2.3 . descreve várias técnicas que possibilitam a estimativa da localização, entre elas: triangulação, proximidade e análise do cenário. Na secção 2.4 . é apresentado um resumo das vantagens e desvantagens de cada técnica e em 2.5 . são referidos algoritmos de localização, focando apenas as redes neuronais.

2.1 . Introdução

Na Imagem 2.1 é apresentada uma proposta de taxonomia para sistemas de

localização baseada no estudo da arte efectuado, tendo como objectivo uma descrição detalhada sobre as características, tipo de sinal e técnicas usadas para a estimativa da localização.

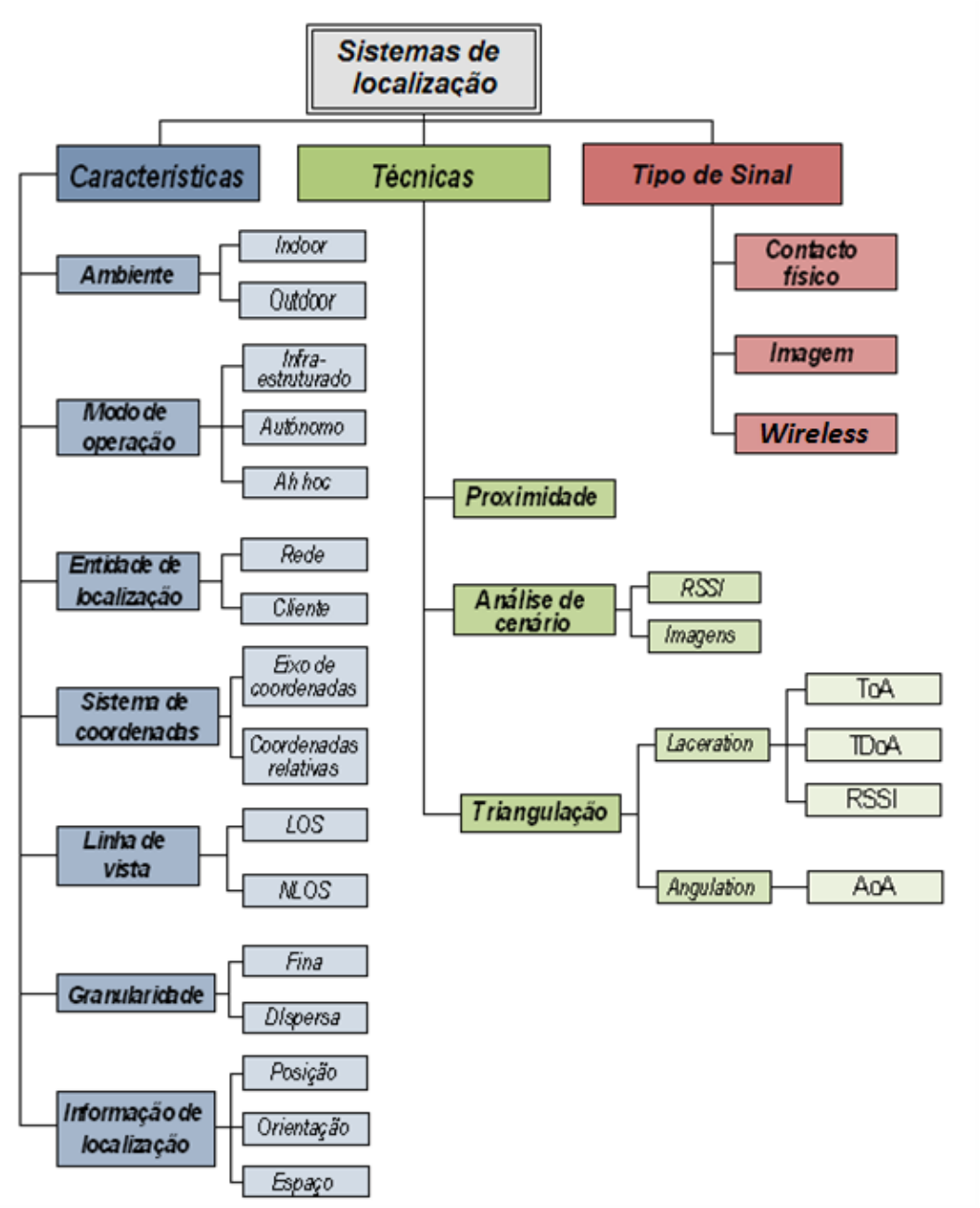

Imagem 2.1: Taxonomia de sistemas de localização

2.2 . Características

Os sistemas de localização ou de posicionamento podem ser caracterizados baseados em vários factores como por exemplo: o ambiente de operação, modo de operação, entidade responsável pela estimativa da localização, sistema de coordenadas utilizado, granularidade, linha de vista e informação de localização. Estas características serão detalhadas nesta secção.

2.2.1 Ambiente de operação

Os sistemas de localização podem ser projectados para operar em diferentes ambientes, "dentro de portas" (*indoor)* ou "ao ar livre" (*outdoor).*

Os sistemas *outdoor* são geralmente baseados em infra-estrutura complexas e dispendiosas, como por exemplo o *GPS, GLONASS* ou baseados na própria rede celular. Estes sistemas têm uma área de cobertura muito abrangente e usualmente não têm restrições de consumo energético nos satélites ou estações base de localização (*Localization Base Station - LBT*) [10].

Os sistemas de localização *indoor* abrangem uma área de cobertura bastante reduzida, comparativamente a sistemas *outdoor* e usualmente têm restrições de consumo energético. Estes ambientes estão sujeitos à presença de variadíssimos obstáculos e interferências, que atenuam ou impedem a propagação do sinal *RF* (rádio-frequência)**.**

2.2.2 Modos de operação

Em relação aos modos de operação podemos dividir o sistema como infra-estrutura, *ad hoc* e autónomo [11].

***Modo infra-estrutura* –** modo de operação onde a estimativa da localização necessita o auxílio de uma infra-estrutura ou de um controlo central. São utilizados pontos de referência (*landmarks*) para auxiliar o processo de localização dos sensores móveis. Neste modo os dispositivos localizáveis não comunicam entre si directamente, necessitam de comunicar com a infra-estrutura e esta comunicará com o dispositivo desejável. Um bom exemplo destes sistemas é a telefonia celular, onde a comunicação entre dois equipamentos móveis tem previamente que passar pela estação base (*Base Tranceiver Station - BTS*) mesmo que os dispositivos móveis se encontrem próximos.

Modo ad hoc **–** modo de operação no qual a estimativa do posicionamento é realizado sem uma infra-estrutura ou um controlo central. Os dispositivos partilham as informações recebidas pelos seus sensores com os dispositivos vizinhos possibilitando assim a troca de informações essenciais para a estimativa da localização. Os dispositivos podem ser localizados de forma relativa em relação a outros dispositivos ou de um modo absoluto quando existem dispositivos de referência cuja posição é previamente conhecida. Este modo é usualmente utilizado em redes de sensores [12].

Modo autónomo **–** modo de operação onde o dispositivo móvel se auto-localiza relacionando a posição anterior conhecida, com as alterações da distância percorrida, direcção e velocidade. Este método é designado por *dead reckoning*. Estes sistemas são bastante usados em sistemas de navegação para robôs em ambientes onde não é desejável ou possível o auxílio de uma infra-estrutura para o suporte à localização. Estes sistemas são muito úteis para explorações planetárias e acções militares [13].

2.2.3 Entidade de localização

A entidade de localização refere-se ao responsável pelo cálculo da estimativa de localização. Poderá ser o próprio dispositivo como é o caso dos módulos *CC2431* que calculam a sua localização através do *location engine [6],* ou poderá ser um ou mais elementos que compõem a infra-estrutura do sistema de localização.

Na Imagem 2.2(a) o próprio dispositivo calcula a estimativa da sua posição através do processamento das informações emitidos pelos dispositivos vizinhos. Estes dispositivos vizinhos podem enviar ciclicamente tramas com informação, como por exemplo a sua identificação e localização. O dispositivo localizável, baseando-se num determinado parâmetro, por exemplo: potência do sinal, tempo de chegada, ângulo de chegada e a posição dos nós de referência estima a sua própria posição. Este modo é adequado quando se pretende manter um nível elevado de privacidade, onde a rede não possui qualquer indicação sobre a posição do dispositivo que calcula a localização.

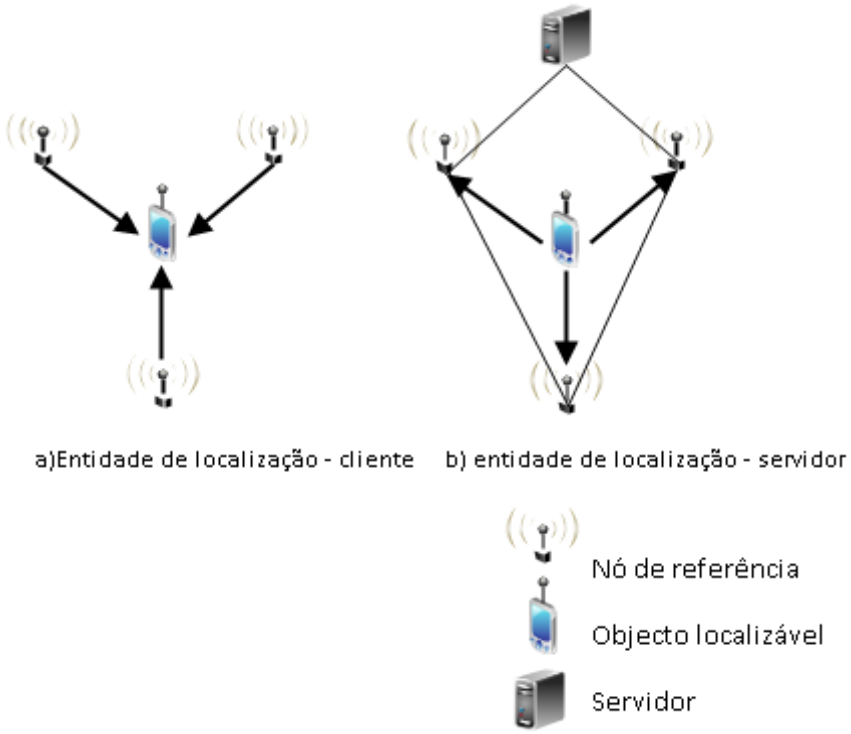

Imagem 2.2: Entidades responsáveis pela estimativa da localização

Na Imagem 2.2(b), o responsável pelo cálculo da estimativa da localização do dispositivo é um servidor. O objecto localizável envia *broadcasts* ciclicamente, estes são detectados pelos nós de referência da infra-estrutura e posteriormente processados ou apenas reencaminhados para um servidor que executará a estimativa da localização do objecto de acordo com a informação recebida. Este modo não permite a privacidade dos dispositivos localizáveis mas por sua vez possibilita-lhes um menor processamento de dados.

2.2.4 Sistema de coordenadas

Para qualquer processo de localização é essencial definir um sistema de coordenadas. Este poderá relacionar as distâncias ou ângulos em relação a um eixo fixo de coordenadas ou através de um eixo de coordenadas relativas entre dispositivos [11].

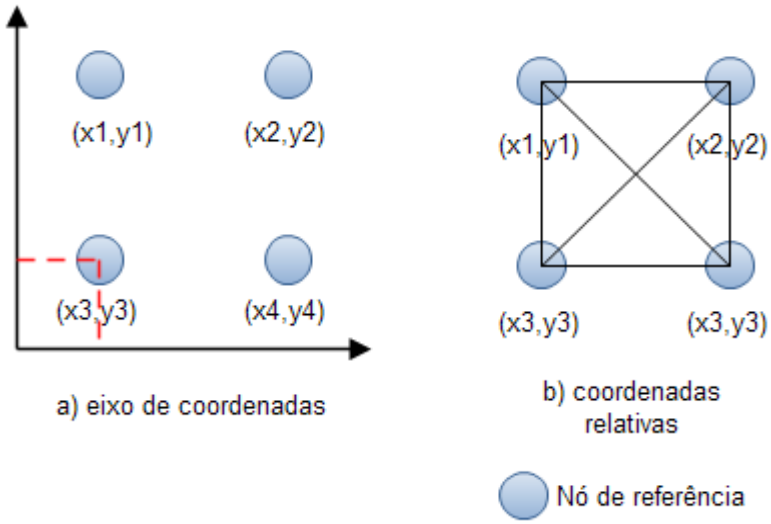

Imagem 2.3: Sistema de coordenadas

A Imagem 2.3 a) diz respeito ao primeiro sistema de coordenadas referido, sendo usualmente utilizado em sistemas infraestruturados onde as medições de distância ou ângulo são feitas em relação a um eixo de coordenadas. No segundo caso realizam-se apenas medições relativas entre os próprios nós, demonstrado pela Imagem 2.3 b).

2.2.5 Modos de representação da localização

Os sistemas de localização podem representar a estimativa da localização de três modos: espaço, posição e orientação [11]. O espaço diz respeito a uma região com limites previamente estabelecidos como uma cozinha, sala, garagem. A posição diz respeito a coordenadas espaciais do dispositivo que se pretende localizar, habitualmente bidimensionais. A informação de orientação pode ser definida como o ângulo entre a direcção onde se encontra o dispositivo e um eixo fixo, dentro de um sistema de coordenadas, podendo ser local ou global. Uma sistema de localização numa residência habitualmente faz uso de um sistema de orientação local, onde são fornecidas informações em relação a um eixo de referência usada apenas por esse local específico.

Imagem 2.4: Modos de localização

2.2.6 Granularidade

As principais características de avaliação de desempenho de um sistema de localização são a exactidão e a precisão [11]. A primeira característica diz respeito à distância da posição estimada no processo de localização com a posição real do dispositivo. A segunda característica diz respeito à dispersão desse valor estimado em relação à posição real, habitualmente referido como a percentagem com que a exactidão referida se verifica ao longo das amostras. Quando se fala de um sistema de localização devem ser sempre referenciados estes dois parâmetros, sendo habitualmente referenciado como: o erro do sistema é de M (metros) em P% (percentagem) das medições.

A granularidade pode então ser definida como o nível de qualidade da exactidão de um sistema de localização. Sistemas *indoor* habitualmente exigem sistemas com granularidade fina, ou seja, nível elevado de exactidão. Sistemas *outdoor* habitualmente não têm essa exigência, dando indicações mais dispersas, como por exemplo, informação de que um dispositivo se encontra no Jardim Municipal do distrito de Aveiro. Para todos os tipos de granularidade, a aplicação é melhorada de acordo com uma maior precisão.

2.2.7 Linha de vista

A linha de vista diz respeito ao percurso do sinal desde o transmissor ao receptor. Quando é possível unir o transmissor e o receptor com uma linha imaginária recta, refere-se que a comunicação é realizada com linha de visão *(Line of Sight - LOS)*. Caso haja

obstrução na comunicação entre o transmissor *Tx* e o receptor *Rx* refere-se que a comunicação é realizada sem linha de visão *(Non-line-of-sight ou Near-line-of-sight - NLOS)*.

Conforme a natureza do sinal usado poderá ser essencial uma comunicação por *LOS* para o funcionamento do sistema, como é o caso de sistemas baseados em infravermelhos ou imagens. Nestes casos a localização está fortemente dependente do dinamismo do ambiente de localização, onde uma alteração da mobília poderá por fim a todo o processo de localização. Os sistemas baseados em sinais de rádio-frequência são mais robustos para estas situações. Estes, conforme a natureza dos elementos obstrutores podem apenas sofrer degradação do sinal mas possibilitar mesmo assim a comunicação entre os dispositivos.

2.2.8 Classificação do tipo de sinal

O sistema de localização pode utilizar diferentes tipos de sinal para o cálculo da posição estimada de um dispositivo. Existem três tipo de sinal habitualmente usados para sistemas de localização [10]: imagem, contacto físico e sinais *wireless*.

- **Imagem**

Esta variante de sistemas de localização baseia-se no processamento de imagens captadas por várias câmaras ligadas a um servidor. O processamento de várias imagens segundo diferentes ângulos através de algoritmos e técnicas avançadas permitem identificar e calcular a posição de pessoas ou objectos. Um dos principais sistemas de localização baseado em processamento de imagens é o *Easy Living* [14]. Este tipo de sistemas de localização implica um grande esforço computacional para a obtenção de uma boa exactidão, principalmente na condição de presença de um número elevado de pessoas ou objectos.

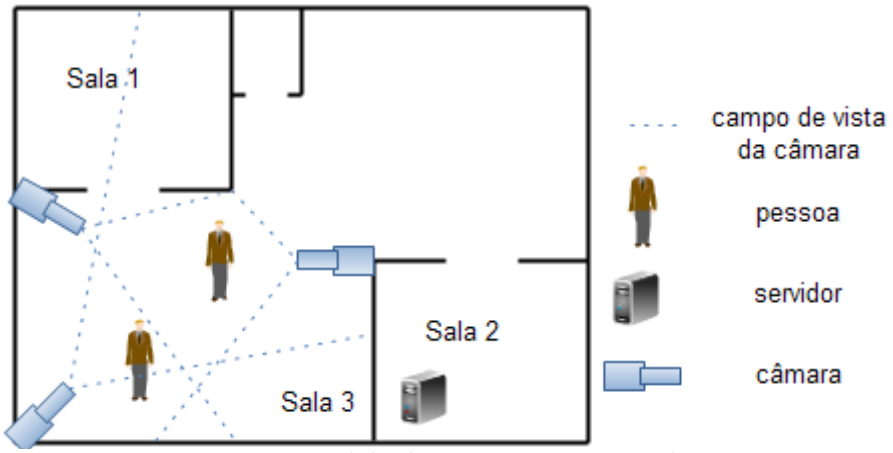

Imagem 2.5: Sistema de localização por processamento de imagens

. Contacto físico

Sistemas de localização baseados em contacto físico estão relacionados no contacto directo de uma pessoa ou objecto com um componente da infra-estrutura do sistema, realizando assim, uma estimativa directa sobre a localização. Podemos referir o sistema de localização *Smart Floor* [4], onde vários sensores de pressão espalhados pelo chão criam um "tapete de localização". Através de uma base de dados com um modelo baseado no peso e inércia de cada utilizador, é possível identificar não só a posição mas também a pessoa que efectuou o contacto directo com os sensores.

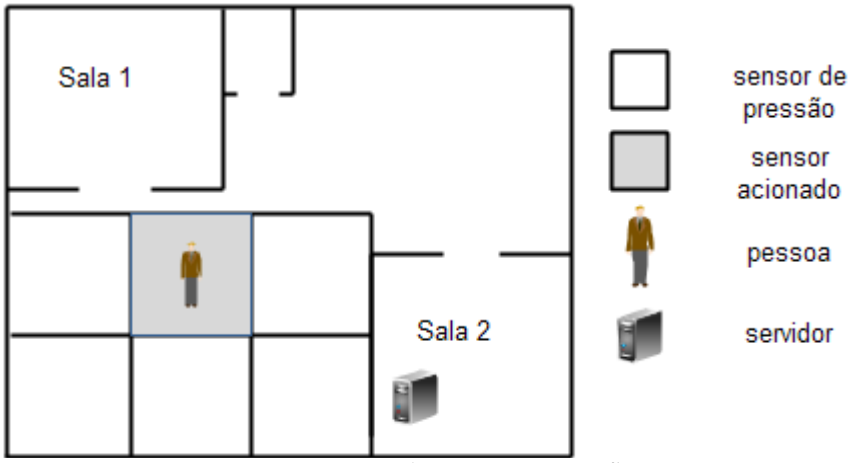

Imagem 2.6: Localização por contacto físico

. *Sinais Wireless*

Os sinais utilizadas em sistemas de localização *indoor* podem ser divididos em três grandes grupos consoante a frequência utilizada: infravermelhos, sinais de rádio e ultra-sons.

. Infravermelhos

Consiste num sinal electromagnético com frequência de onda compreendida entre 300 *GHz* e 400 *THz* [16]. Este sinal é radiado fortemente por todos os corpos que geram ou armazenam calor. Sinais infravermelhos possibilitam uma boa exactidão para sistemas de localização mas estão sujeitos a interferências de luminosidade do ambiente. O sinal infravermelho não atravessa a maioria dos obstáculos, tais como, paredes e mobília, inerentes a qualquer sistema *indoor.*

Apresentam limitações no alcance, cerca de 5 metros, mas por outro lado os transmissores infravermelhos são baratos, compactos e de baixo consumo energético. Um dos primeiros sistemas de localização a adoptar esta tecnologia foi o *Active Badge* [17].

. Rádio frequência

Os sinais *RF* consistem em ondas electromagnéticas com frequência inferior à radiação infravermelha. O alcance deste tipo de sistemas depende da potência de transmissão e da atenuação que o sinal sofrer no ambiente. A sua velocidade de propagação é bastante elevada, aproximada à velocidade da luz $(3\text{x}10^8 m/s)$ possibilitando uma latência bastante atractiva. O facto de haver gamas de frequência não licenciadas disponíveis para o uso deste tipo de sinal também o faz bastante atractivo. Alguns exemplos de sistemas que utilizam tecnologia baseada em sinais de rádio-frequência são o *RADAR[19], LANDMARC[20], Horus[21] e Ubisense[37].*

. Ultra-sons

O ultra-som é um sinal que opera em bandas de frequência superiores às que o ouvido humano pode distinguir (20*kHz*), podendo ir até centenas de *kHz*.

Os ultra-sons não conseguem penetrar paredes ou obstáculos e a sua propagação é influenciada pela temperatura do ambiente. O alcance é reduzido, entre 3 a 10 metros mas possibilita aos sistemas de localização uma resolução em torno dos cms. Para

sistemas de localização que utilizam esta tipo de sinal podemos referir o *Active bat* [21] e o *Cricket*[11].

2.3 . Técnicas de localização

Os sistemas de localização podem ser categorizados segundo diferentes técnicas, as quais são usadas para estimar a localização dos dispositivos móveis. Não existe até hoje nenhum modelo ideal para descrever as características de multi percurso inerentes a sistemas de localização *indoor,* existem apenas técnicas que possibilitam uma aproximação dos valores reais. Estas podem ser categorizadas em três grandes tipos: triangulação, proximidade e análise do cenário [11].

2.3.1 Triangulação

Esta técnica baseia-se nas propriedades geométricas dos triângulos e pode ser baseada numa técnica baseada na distância (*lateration)* ou na diferença angular (*angulation*) entre dispositivos. Através da medição do tempo de propagação do sinal entre o emissor e vários receptores, e conhecendo previamente a velocidade de propagação deste, é possível estimar a distância do dispositivo localizável a vários pontos de referência. Do mesmo modo, medições do nível de potência recebido, *RSSI*, permitem calcular a distância percorrida do sinal relacionando a potência de emissão e recepção com um modelo de perdas de propagação. Através da obtenção de pelo menos três distâncias é possível estimar a posição do dispositivo. Ambos estes casos se enquadram na técnica *lateration [7]*. Na técnica *angulation*, é efectuado o cálculo posicional baseado numa medição angular, onde se estima a posição relacionando a direcção de propagação entre o objecto e vários pontos de referência.

. Tempo de chegada (*ToA*)

Como o nome indica, esta técnica baseia-se na medição do tempo de chegada de um sinal entre o transmissor e o receptor. Através da cronometragem do tempo de percurso (*time of flight - ToF)* entre a transmissão do sinal e sua recepção no elemento receptor é possível calcular a distância entre os dois dispositivos.

Nesta técnica é necessário um sincronismo muito preciso entre o transmissor e o receptor, especialmente se o sinal usado para a comunicação for *RF*. Neste caso, um

desvio de *1μs* na medição do tempo poderá impor erros de centenas de metros.

A determinação da posição segundo esta técnica poderá ser realizada pelo controlar da rede ou pelo dispositivo móvel.

No primeiro caso o controlador envia um pedido de localização para a rede, isto possibilita aos dispositivos de referência a inicialização de uma escuta sincronizada. O dispositivo localizável ao receber o pedido de localização responde com um sinal que será recebido pelos dispositivos de referências. Os diferentes dispositivos de referência irão calcular a diferença temporal entre o pedido de localização e a resposta do dispositivo móvel que corresponderá ao tempo de chegada do sinal. Estes dados serão enviados para o controlador que, obtendo no mínimo três tempos de chegada de dispositivos de referência não colineares, poderá calcular a distância destes ao dispositivo móvel através do conhecimento prévio da velocidade de propagação. Para uma obtenção de valores mais precisos poder-se-à subtrair a esta diferença temporal o tempo de processamento do dispositivo móvel desde a recepção do pedido até à resposta deste.

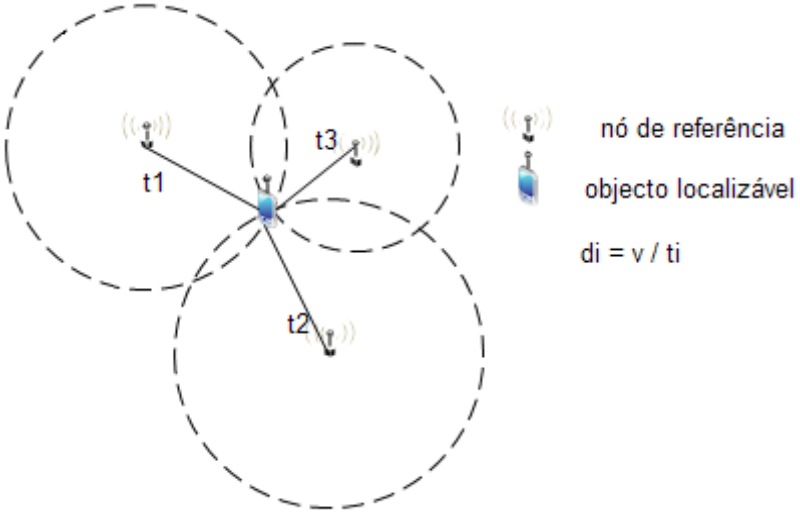

Imagem 2.7: Representação da técnica ToA

Na Imagem 2.7 é representada a técnica *ToA*, onde *t1* a *t3* representam o *ToA* entre os dispositivos de referência e o dispositivo cuja localização se pretende estimar. Sabendo a velocidade de propagação do sinal é possível calcular a distância entre eles. Esta distância é definida como o raio da esfera centrada num respectivo ponto de referência. A intercepção das esferas (pelo menos três) possibilita a estimativa da posição do dispositivo. O *Active Bat* [21] é um sistema que utiliza esta variante de *ToA*.

No segundo caso, onde o cálculo é efectuado pelo dispositivo móvel, é realizado um envio periódico de mensagens por parte dos dispositivos de referência. Estes enviam uma mensagem indicando o tempo em que foi iniciada a emissão do sinal e a sua posição.

O dispositivo móvel, ao receber esta mensagem de diferentes dispositivos de referência e sabendo a velocidade de propagação do sinal consegue estimar a distância a que se encontra dos dispositivos de referência. A partir deste momento, através do uso de cálculos geométricos é possível estimar a posição do dispositivo móvel.

Existe uma variante desta última técnica, designada *RTOF* (*Roundtrip time of flight*). Nesta técnica o dispositivo mede o intervalo de tempo entre o envio do seu sinal e a recepção da resposta dos dispositivos de referência, não sendo necessário o envio na mensagem do tempo em que foi realizada a transmissão do sinal. Esta técnica funciona como um radar comum.

. Diferença do tempo de chegada (*TDoA*)

A técnica diferença do tempo de chegada (*TDoA)* baseia-se na diferença temporal entre a recepção de vários sinais emitidos sincronizadamente num único dispositivo ou na diferença temporal da recepção de vários dispositivos de referência de um único sinal transmitido. Para o segundo caso referido, o dispositivo móvel emite um sinal para uma rede de sensores, sendo calculada a diferença temporal entre os dispositivos de referência através das equações [7]:

$$T_A = 1/c\sqrt{(x - x_A)^2 + (y - y_A)^2}\,(s)$$

$$T_B = 1/c\sqrt{(x - x_B)^2 + (y - y_B)^2}\,(s)$$

$$T_C = 1/c\sqrt{(x - x_C)^2 + (y - y_C)^2}\,(s)$$

$$\tau_B = T_B - T_A = 1/c\left(\sqrt{(x - x'_B)^2 + (y - y'_B)^2} - \sqrt{(x^2 + y^2}\right)\,(s)$$

$$\tau_C = T_C - T_A = 1/c\left(\sqrt{(x - x'_C)^2 + (y - y'_C)^2} - \sqrt{(x^2 + y^2}\right)\,(s)$$

onde $T_A, T_B e T_C$ representam o *ToA* do sinal aos dispositivos de referência A,B e C. $\tau_B e \tau_C$ a diferença temporal entre os dispositivos BA e CA; $x'_B e y'_B$ a diferença posicional entre os dispositivos B e A; $x'_C e y'_C$ a diferença posicional entre os

dispositivos C e A; x e y posições que pretendemos calcular; c a velocidade de propagação do sinal. Através de duas equações é possível resolver o sistema e estimar o valor da posição (x, y) . Se a posição estimada for representada em coordenadas tridimensionais necessitar-se-ia de três equações de τ .

As equações da diferença temporal representam um hiperbolóide entre os dois dispositivos de referência, sendo possível estimar a posição do objecto através da intercepção de duas ou mais hiperbolóides. Este método também é conhecido por *hyperbolic position location [7]*.

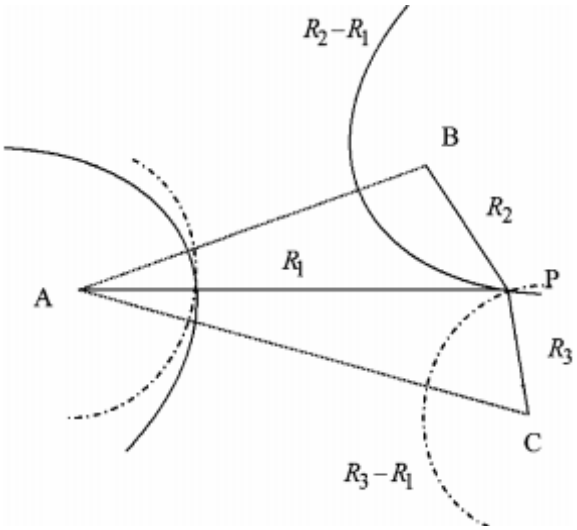

Imagem 2.8: Representação da técnica TDoA[7]

Como exemplo de sistemas que utilizam a técnica *TDoA* são de referenciar o sistema *60 GHz OFDM* [23] e o sistema *Cricket [11]*.

. Received Signal Strengh Indicator *(RSSI)*

Este método recorre à análise da intensidade do sinal recebido, *RSSI*, no receptor e a um modelo que descreve como esta varia com a distância.

Existem vários modelos de aproximação para estimar as perdas de propagação da intensidade do sinal. Estes podem ser subdivididos em modelos empíricos (estatísticos) ou teóricos (determinísticos), caso se baseiem em medições efectuadas no ambiente de localização ou nos fundamentos de propagação de ondas [24].

Os modelos empíricos têm como base as medições realizadas no ambiente de localização, tomando implicitamente em atenção todas as influências do ambiente no momento em que foi realizada a análise deste. Deste modo é efectuada uma calibração

prévia do sistema onde através da análise do ambiente de localização são estimados os parâmetros que melhor descrevem o ambiente de localização.

Os modelos determinísticos são baseados em princípios físicos de modo a analisar a propagação do sinal. Estes modelos requerem uma completa base de dados sobre as características do ambiente, o que na maioria dos casos é impraticável. Apesar de, quando implementados de forma correcta, ser possível uma melhor eficiência do que com modelos empíricos.

O módulo *CC2431* usado nesta dissertação inclui um *Location Engine* que através da equação descrita em seguida permite uma calibração de acordo como o sistema de localização considerado [6].

$$RSSI = -(10 \times N \times \log_{10} d + A)(dBm)$$

N – constante de propagação do sinal;

d – distância ao emissor;

A – intensidade do sinal a uma distância de 1 metro;

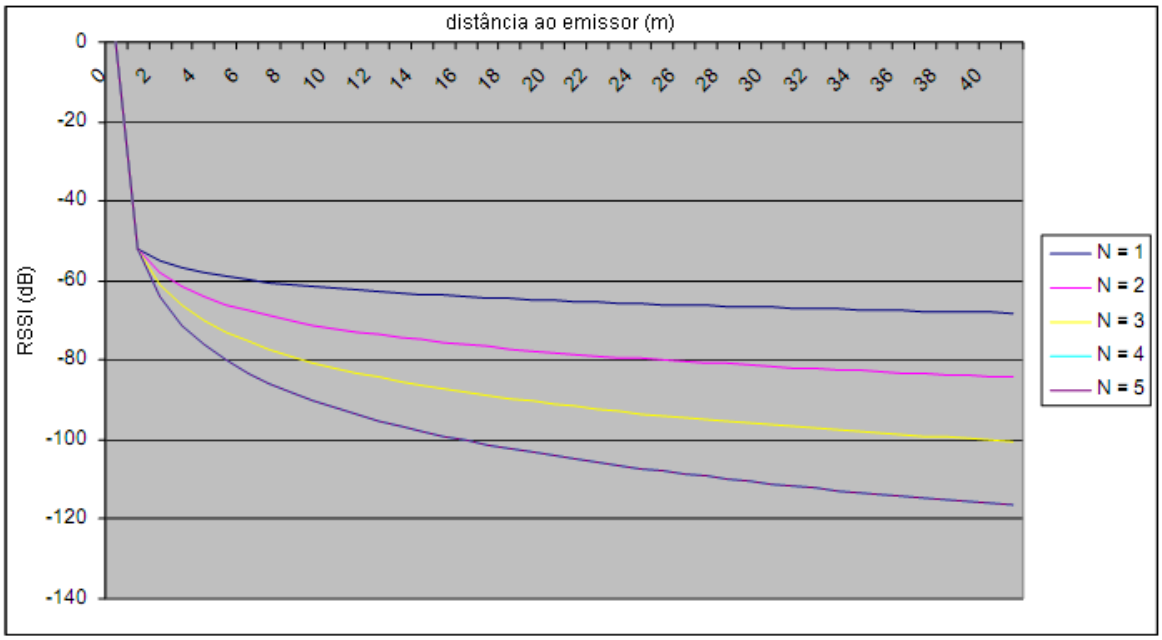

Imagem 2.9: Modelo de perdas de propagação do módulo CC2430/31 [6]

De acordo com uma análise empírica podem-se configurar os valores de N e A que melhor descrevam o comportamento da potência do sinal no ambiente de localização.

. *Angulation*

Este método utiliza a diferença angular entre dispositivos para calcular a localização destes. *AoA (Angle of Arrival)* é definido como o ângulo entre a direcção de propagação do sinal incidente e uma direcção de referência, conhecido como orientação. A orientação é representada em graus no sentido do relógio desde o Norte. Quando a orientação é de 0^o ou apontada para o Norte, o *AoA* é absoluto, de outro modo é relativo. Um modo comum usado para obter o *AoA* é usar um vector de antenas ou antenas direccionais em cada nó de referência para calcular o ângulo de chegada dos sinais emitidos pelo dispositivo localizável. Estes dispositivos de referência têm um eixo principal que é usado como referência para medir os ângulos de chegada [29].

Através do conhecimento prévio da distância entre os vários receptores e dos ângulos de chegada detectados em cada ponto de referência, é possível estimar a posição do objecto através de funções trigonométricas.

Na Imagem 2.10 é demonstrada uma localização baseada em *AoA* onde a entidade de localização é o cliente, o próprio nó cuja posição se deseja calcular.

Em a) a orientação do dispositivo localizável é conhecida, o que permite calcular o ângulo absoluto em relação aos nós de referência *b1* e *b2*. Deste modo é possível restringir a posição do dispositivo móvel a uma linha de início em cada nó de referência. Neste caso duas medidas angulares de dois nós diferentes não colineares seriam necessários para estimar uma localização a 2D ou três medições para estimar uma localização a 3D [29].

No caso de b) a informação do ângulo absoluto não é conhecida logo a estimativa da posição é efectuada através da diferença de *AoA* de diferentes nós de referência.

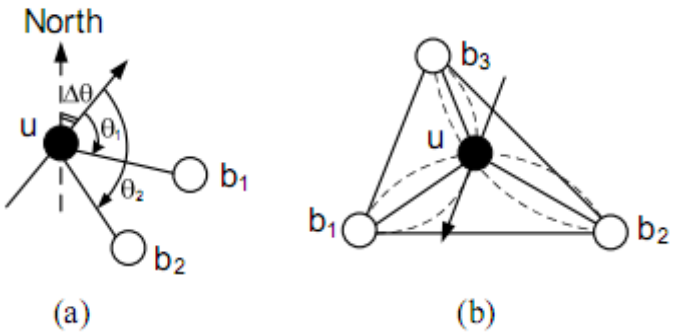

Imagem 2.10: Técnica AoA a) AoA com orientação conhecida b) AoA sem orientação conhecida[29]

Tanto em a) e b) o conhecimento da posição dos nós de referência é essencial para o

processo de localização.

Um problema desta técnica diz respeito à necessidade de vectores de antenas ou de antenas direccionais o que não é muito usado em terminais móveis devido ao custo. Outro grande problema refere-se ao efeito de propagação por múltiplos caminhos (*multipath)*. Pode ocorrer a situação em que o sinal registado não é o sinal em *LOS*, mas sim um sinal reflectido, o que levaria a uma informação incorrecta sobre a posição do dispositivo a localizar [7].

Também existem sistemas híbridos como por exemplo *ToA/AoA e TdoA/AoA*, o que permite aumentar a exactidão de uma estimativa de localização para diferentes cenários. Alguns desses sistemas são referenciados em [25] [26].

2.3.2 Proximidade

Esta técnica está relacionada com a detecção de objectos através da sua proximidade com os dispositivos de referência. Nesta técnica apenas é identificado que o dispositivo foi localizado próximo do dispositivo de referência mas não é conhecido a distância exacta entre os dispositivos. Quando um dispositivo é detectado por um único nó, a localização do dispositivo é considerada próxima desse nó. Caso a detecção seja efectuada por mais de um nó de referência a posição é atribuída ao nó que detectar maior potência de sinal, como é o caso de um dos modos de localização do sistema implementado. Sistemas baseados em infravermelhos e *RFID (radio frequency identification)* usualmente utilizam esta técnica [7].

2.3.3 Análise do cenário

Esta técnica baseia-se nas características únicas de cada posição do cenário de localização que pretendemos observar. O cenário pode ser observado através de imagens [14], ou através da intensidade do sinal recebido, *RSSI*.

O processo de estimativa de localização através desta técnica divide-se em duas fases, *offline* e *online*. Na primeira fase, também conhecida por fase de calibração, é realizado um mapa com os diferentes arranjos de *RSSI* de uma série de receptores para cada uma das posições, sendo guardados posteriormente numa base de dados. Estes padrões para cada posição são designados por *fingerprints* ou assinaturas *RF* [28].

A fase *online* refere-se ao momento do processo de localização. Este processo irá ser

executado através do processamento dos valores de potência recebidos segundo um determinado algoritmo de localização com base nas assinaturas *RF* guardadas na base de dados. O parâmetro de saída deste algoritmo será a posição estimada do dispositivo.

A grande desvantagem deste sistema refere-se ao tempo gasto para efectuar a calibração da rede, obrigando a novas calibrações sempre que haja uma reorganização da área de localização.

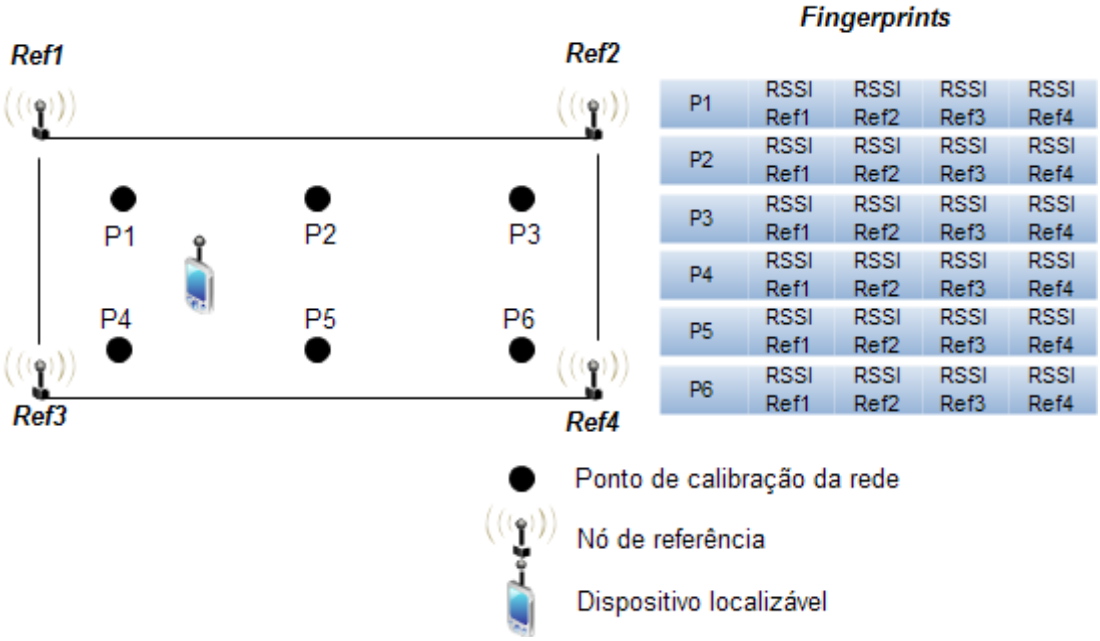

Imagem 2.11: Representação de um sistema baseado em análise de cenário por RSSI

Esta técnica de localização está directamente associada a um algoritmo de localização, que de acordo com os *arrays* de *RSSI* recebidos possibilita a estimativa da localização do dispositivo. O uso de uma calibração e do suporte com algoritmos de localização permitem minimizar erros devido às variações de *RSSI,* isto porque são calibradas de acordo com o sistema e ambiente de localização.

. Factores de variação de *RSSI*

Existem muito factores que influenciam o valor de *RSSI* em ambientes *indoor*. Entre eles podemos mencionar: variabilidade do transmissor e receptor; o material e organização da sala de localização; presença de pessoas o que está directamente relacionado com o tempo do dia; orientação e directividade das antenas; multi percurso e desvanecimento do canal *(fading)* [28].

Variabilidade do transmissor – transmissores diferentes podem comportar-se de modo diferente mesmo quando estão configurados exactamente do mesmo modo. Isto significa que quando um transmissor é configurado para enviar sinais com uma determinada potência então o transmissor irá transmiti-los a uma potência muito próxima mas não necessariamente igual à configurada. Isto altera o valor de *RSSI* recebido podendo assim levar a estimativas de distância incorrectas.

Variabilidade do receptor – a sensibilidade dos receptores de rádio pode ser diferente. Isto significa que o valor de *RSSI* medido em diferentes receptores pode ser diferente mesmo quando todos os outros parâmetros que afectam o *RSSI* são mantidos constantes.

Orientação da antena – cada antena tem o seu próprio diagrama de radiação que não é uniforme. Isto significa que o valor de *RSSI* medido no receptor para um par de nós de comunicação e uma dada distância varia conforme a orientação das antenas do receptor e transmissor.

Multi percurso e desvanecimento do canal – num ambiente *indoor* o sinal transmitido é reflectido e refractado em paredes e/ou outros objectos. Além das diferentes atenuações que os diferentes materiais podem provocar no sinal original, o sinal recebido no receptor poderá ser originário de um sinal reflectido o que poderá levar a uma interpretação errada na localização.

2.4 . Comparação de técnicas de localização

Para uma mais fácil compreensão das vantagens e desvantagens de cada técnica é descrito na Tabela 2.1 um resumo sobre as características das três técnicas de localização referidas anteriormente.

	Vantagens	Desvantagens
ToA/ TDoA	*Elevada exactidão para condições LOS* *Não necessita de treino exaustivo* *Boa escalabilidade*	*Elevados requisitos de sincronização* *Hardware complexo (preço mais elevado)* *Maiores requisitos de largura de Banda* *Muito sensível a condições NLOS*
AoA	*Necessidade de apenas 2 medidas para localização a 2D e 3 para 3D* *Não necessita de elevados requisitos de sincronização* *Não necessita de treino exaustivo*	*Hardware complexo (preço mais elevado)* *Elevados problemas em condições NLOS*
RSSI **fingerprinting**	*Hardware simples* *Sem custos adicionais de sincronismo* *Maior resiliência para condições NLOS* *Menor sensibilidade à largura de banda*	*Menor exactidão do que sistemas baseados em ToA ou AoA em condições de LOS* *Elevado tempo de treino* *Algoritmos de elevada complexidade* *Não apresentam elevada escalabilidade*

Tabela 2.1: Comparação de técnicas de localização

2.5 . Algoritmos de localização

Existem vários algoritmos de localização baseados em *fingerprinting*, entre eles: métodos probabilísticos (*inferência Bayesiana [32] e de Dempster-Shafer*); detecção por vizinhança mais próxima (*k-nearest neighbor [18] – kNN*); redes neuronais [30]; máquina de suporte vectorial (SVM) *[31];* e menor polígono M-vertex *(SMP)* [7].

Estes algoritmos estimam a localização baseando-se na calibração da fase *offline*. Nesta fase são registadas assinaturas *RF* associadas a posições específicas. Esta calibração permite posteriormente relacionar os valores de *RSSI* detectado na fase de detecção (*online*) com as assinaturas estimando assim um valor para a posição do dispositivo localizável.

O sistema de localização usado nesta dissertação utiliza um algoritmo baseado em proximidade para atribuir a localização do dispositivo móvel à localização do dispositivo de referência que detectar o seu maior nível de potência. Parte-se do princípio que o dispositivo de referência mais próximo irá detectar o nível de potência mais elevado o que na prática nem sempre se verifica devido às variabilidades anteriormente descrita. Para estes casos o uso de um algoritmo de maior complexidade é essencial para um melhor desempenho de todo o sistema de localização.

Não faz parte do âmbito desta dissertação um estudo aprofundado sobre algoritmos

de localização, deste modo foram apenas referidos vários algoritmos e descrito com maior ênfase o algoritmo usado nesta dissertação, redes neuronais.

. Redes neuronais

Uma rede neuronal artificial tem a sua origem na analogia com uma rede neuronal cerebral onde a unidade análoga ao neurónio é denominada por elemento de processamento (*EP*) ou neurónio artificial.

Os neurónios cerebrais são formados por *dendritos* (terminais de entrada), corpo da célula e por *axónios* (terminais de saída) representados na Imagem 2.12. A região onde dois neurónios entram em contacto e transmitem os impulsos nervosos é designada por *sinapse*.

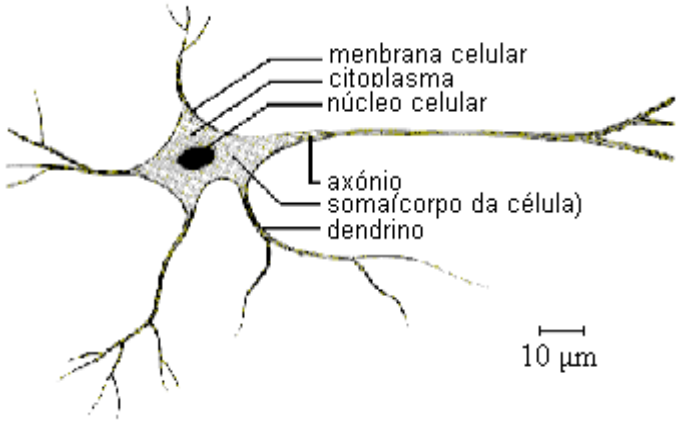

Imagem 2.12: Neurónio humano[30]

O funcionamento dos neurónios pode ser descrito pelo seguinte exemplo: os impulsos recebidos por um neurónio são processados num determinado instante. Se for atingido um determinado limiar, este neurónio produz e envia uma substância neurotransmissora que atravessa o corpo da célula para o *axónio* que poderá estar conectado a um *dendrito* de outro neurónio. Conforme esta substância neurotransmissora pode ser inferido no neurónio receptor uma inibição ou geração de pulsos eléctricos [30].

O funcionamento de um neurónio artificial é idêntico, como verificado na Imagem 2.13. Consiste num conjunto de μ elementos de entrada que são multiplicados por uma matriz de pesos ω e combinados através de uma função Φ. Normalmente a função Φ representa a soma das entradas, mas pode assumir outras funções como por exemplo produto entre entradas, sendo designada esta combinação ponderada das entradas do

neurónio por *confluência* (*x*).

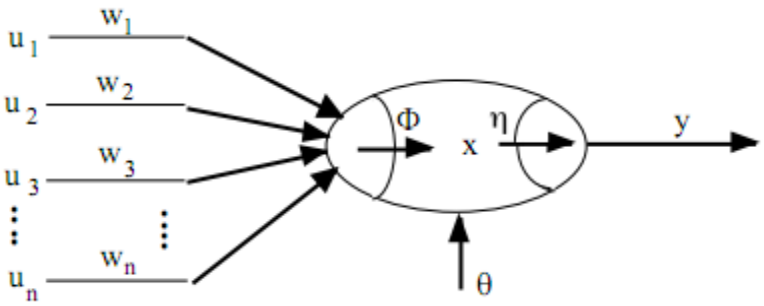

Imagem 2.13: Esquema de um neurónio artificial[30]

Existe um parâmetro auxiliar e opcional designado por *bias (θ)*, que pode ser usado para representar um valor limiar mínimo (*threshold*) para a *confluência* que caso não ultrapasse o limiar será considerado como nulo.

Em seguida esta *confluência* será o parâmetro de entrada de uma função não linear *η* normalmente uma função sigmoidal (variação entre 0 e 1) ou tangente hiperbólica (variação entre -1 e 1), produzindo assim o parâmetro de saída do neurónio, *y*.

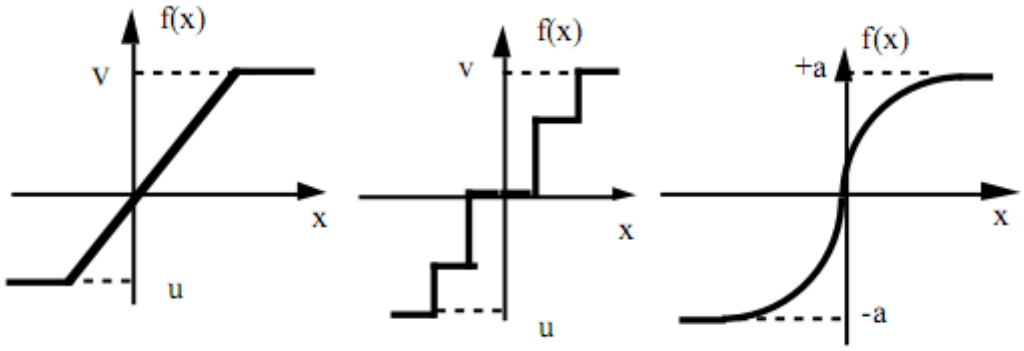

Imagem 2.14: Exemplo de funções não lineares η[30]

Tipicamente a função de transferência é escolhida pelo programador e os parâmetros *ω* e *θ* são ajustados através de alguma regra de aprendizagem de modo a obter uma relação correcta entre os parâmetros de entrada e de saída.

Para modelar funções complexas podem ser interligados múltiplos neurónios e interligados em paralelo de modo a formar camadas de neurónios. Múltiplas camadas de neurónios podem ser ligadas em série resultando numa rede designada por *multi-layer-perceptron (MLP)*. Esta é constituída por uma camada de entrada que recebe os dados; uma camada de saída com a resposta da rede para uma determinada combinação de entradas; camadas de conexão ocultas ou intermediárias (pois não apresentam ligação

com o exterior) que funcionam como funções de transferência entre camadas.

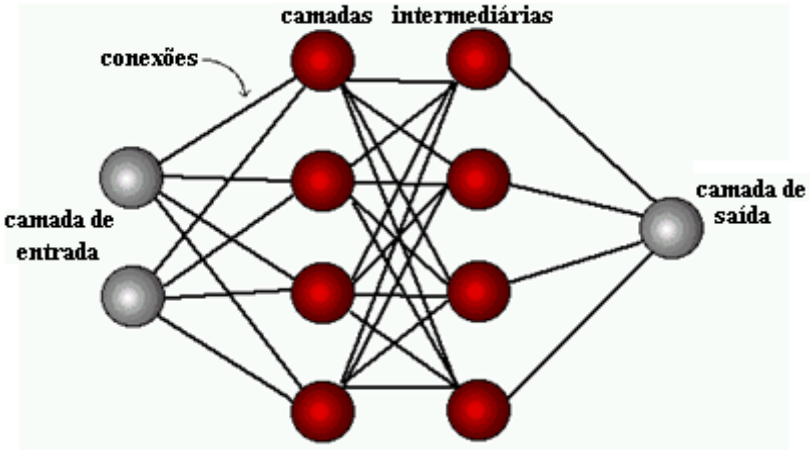

Imagem 2.15: Representação de uma rede neuronal MLP [30]

. Aprendizagem da rede

Para a operação de uma rede neuronal artificial é necessário previamente realizar uma aprendizagem ou treino da rede. Esta aprendizagem pode ser supervisionada caso sejam conhecidos os parâmetros de entradas e os parâmetros de saída ou não supervisionada se apenas forem conhecidos os parâmetros de entrada.

Na aprendizagem supervisionada são inseridos os parâmetros de entrada e os valores esperados da saída, de acordo com estas informações a rede neuronal adapta os valores das matrizes de pesos e valor de *bias* de modo a criar um modelo.

. Algoritmo de aprendizagem

Podem ser implementados vários algoritmos de aprendizagem para a rede neuronal supervisionada tais como: Regra de aprendizagem de *Widrow-Hoff*, aprendizagem por *retropropagação* do erro (*error backpropagation*), algoritmos evolutivos, etc [30].

O algoritmo mais conhecido para treino de redes neuronais é o algoritmo *back-propagation*. Este algoritmo supervisionado utiliza os pares de entrada e de saída esperada para, através de um mecanismo de correcção de erros, ajustar a matriz de pesos. O treino é dividido em duas fases, *forward* e *backward*. Na primeira fase, fase *forward*, os parâmetros de entrada percorrem toda a rede neuronal até serem calculados os parâmetros de saída. Os parâmetros calculados são comparados com os valores esperados calculando assim o erro final. Em seguida é realizada uma computação em

sentido oposto, fase *backward*, onde através do erro final é calculado o erro local para cada elemento da rede, através de um mecanismo de correcção de erros baseado no método gradiente descendente, possibilitando um ajuste da matriz de pesos da rede e de *bias*.

O sistema implementado baseia-se em redes neuronais e o seu funcionamento global pode ser descrito do seguinte modo:

Durante o treino (fase *offline*), os valores de *RSSI* são os parâmetros de entrada (*inputs*) e a posição de localização corresponde ao parâmetro de saída (*output*) da rede neuronal. Através de imensos pares de entrada/saída é possível executar o processo de adaptação dos pesos das conexões ω e *bias*.

Na fase *online*, o vector de entradas *RSSI* apresentado na camada de entrada é transferido para a primeira camada oculta onde cada elemento do vector será multiplicado pelo peso da sua conexão (previamente determinado) e posteriormente somado. Em seguida é aplicada a função tangente hiperbólica e gerado uma saída que será um parâmetro de entrada para outro elemento de processamento da camada oculta seguinte ou o parâmetro de entrada da camada de saída caso seja a última camada oculta.

A saída do sistema será representada como um ponto de duas ou de três dimensões conforme a calibração na fase *offline*.

Em [31] é representado um sistema baseado em redes neuronais.

Capítulo 3

Sistemas de localização

Neste capítulo são descritos alguns sistemas de localização *indoor* baseados nas mais diversas tecnologias assim como o seu modo típico de funcionamento. Existem vários sistemas de localização *indoor* desenvolvidos com o uso das mais diversas tecnologias, entre elas encontram-se: baseadas em contacto físico e análise de imagem; *GPS; RFID*; rede celular; *UWB; WLAN; Bluetooth; ZigBee;* infravermelhos e ultra-sons [7].

Na Imagem 3.1 são descritas tecnologias de localização baseadas em sinais *wireless* assim como a relação alcance/resolução.

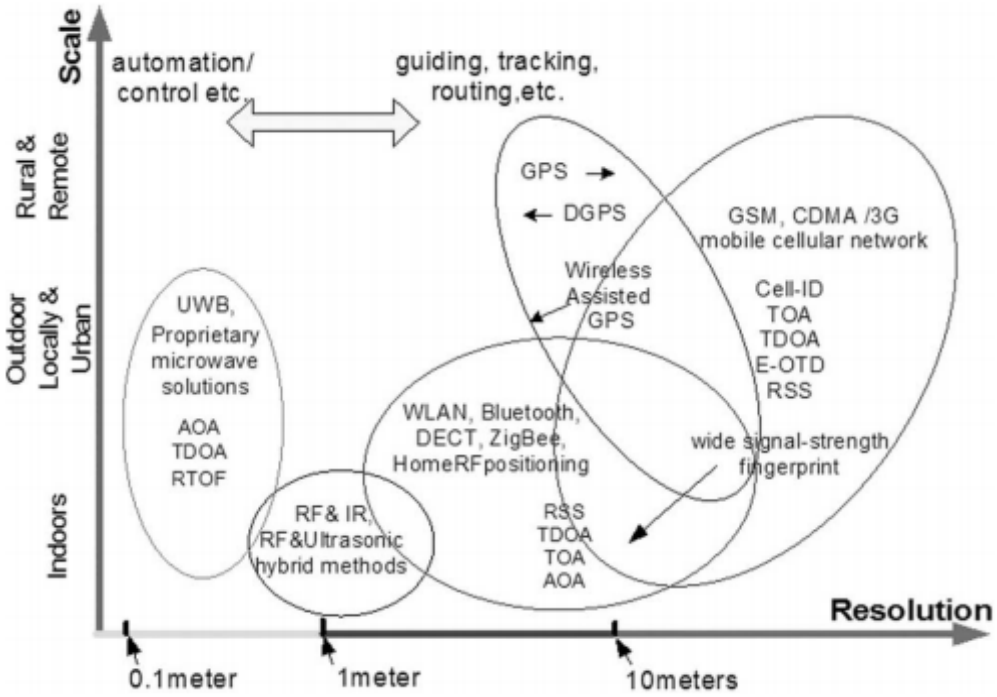

Imagem 3.1: Sistemas de localização baseados em sinais wireless [7]

3.1 . Sistemas baseados em contacto físico

Sistemas de localização baseados em contacto físico estão relacionados com o contacto directo de uma pessoa ou objecto com um componente da infra-estrutura do sistema, realizando assim, uma estimativa directa sobre a localização. O sistema *Smart Floor* [4] é um exemplo de um sistema que se baseia neste princípio.

3.1.1 *Smart Floor [4]*

O *Smart Floor* é um sistema de localização que identifica pessoas através do contacto directo de um utilizador com um conjunto de placas que incorporam células de carga (*load cell*). Dá a ideia de um tapete sensível à pressão que consoante a posição dos utilizadores envia sinais eléctricos para um servidor que processará as informações recebidas.

Este sistema necessita de calibração (fase *offline*) para criar um perfil de localização para cada utilizador baseado no seu peso e inércia.

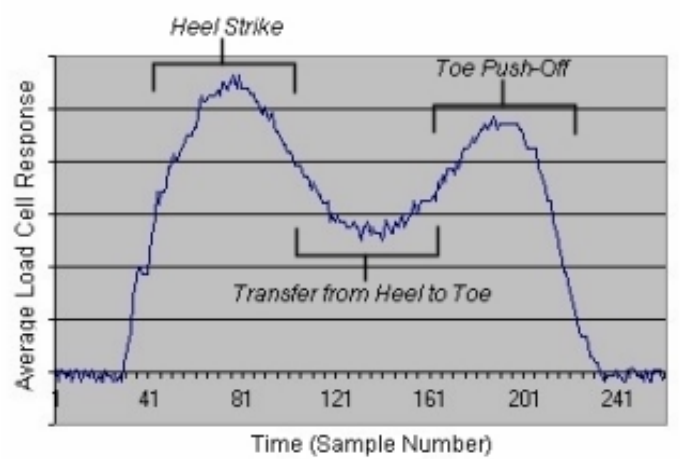

Imagem 3.2: Perfil de um utilizador baseado em GRF[4]

Esse perfil relaciona a forças de reacção ao apoio (*GRF Ground Reaction force*) com o tempo médio para cada pegada. Como se pode verificar pelo gráfico representado na Imagem 3.2, a primeira curva corresponde à força exercida pelo utilizador no momento do apoio traseiro do pé no tapete de sensores. Em seguida segue-se um período de amortização onde ambas as partes frontal e traseira pressionam o tapete, e em seguida segue-se a fase de propulsão que corresponde ao momento em que o utilizador apenas exerce força com a parte frontal do pé.

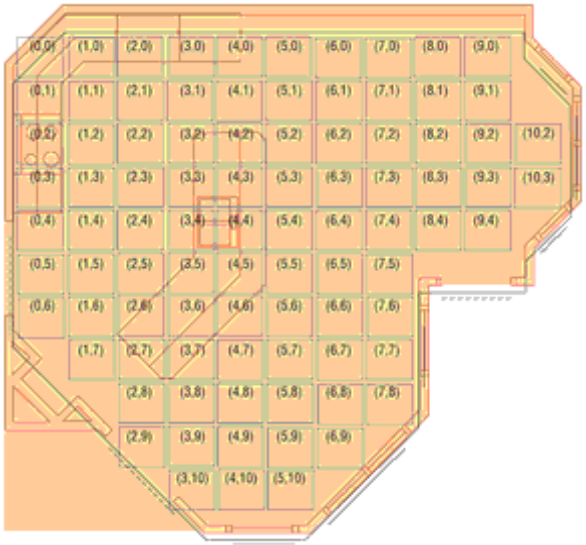

Imagem 3.3: Mapa de posições de um sistema Smart Floor[34]

Os autores de [4] afirmam que o tipo de calçado não influencia significativamente a

localização e que alcançaram 93% de identificação dos utilizadores correcta com 100% de exactidão da área em que os utilizadores estão localizados.

A grande desvantagem deste sistema está relacionada com os custos devido ao elevado número de placas, sensores e à sua manutenção.

3.2 . Sistemas baseados em análise de imagens

Estes sistemas são baseados no processamento de imagens, através de câmaras espalhadas pelo ambiente de localização. Através do processamento de imagens registadas por diferentes câmaras segundo diferentes ângulos é possível estimar a localização e identificação de utilizadores. Um sistema que usa esta técnica é o *Easy Living* [14].

3.2.1 *Easy Living[14]*

Este sistema consiste em duas câmaras estéreo (câmaras com duas ou mais lentes que permitem uma visão binocular) ligadas a dois *PCs* distintos inseridos num ambiente *indoor*, que por sua vez serão ligados a um *PC* que irá executar uma programa de traqueamento.

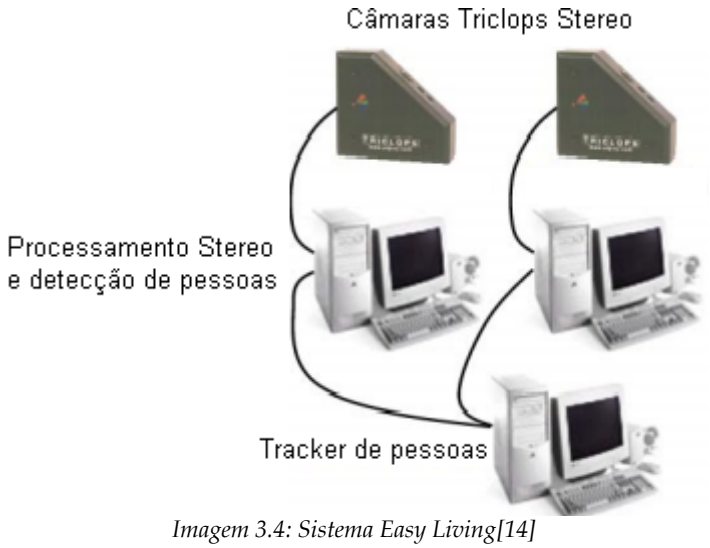

Imagem 3.4: Sistema Easy Living[14]

Cada câmara identifica uma região e envia essas informações para um *PC* que executa um programa de processamento de imagem. Através de uma calibração prévia

para cada câmara, o programa irá subtrair o plano de fundo da imagem recebida e identificar as alterações no campo de visão de cada câmara. As diferentes alterações registadas são conhecidas por *blobs (manchas de cor)*. O corpo de uma pessoa é habitualmente constituído por vários *blobs* que ao serem agrupados permitem a identificação de diferentes pessoas.

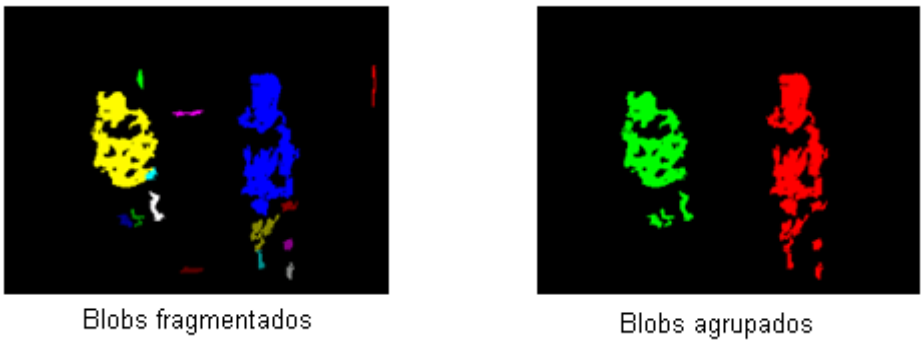

Imagem 3.5: Blobs fragmentados e agrupados [14]

Estes computadores enviam a informação da localização do plano a 2D dos diferentes grupos de *blobs* para um computador a executar um programa designado por *tracker*. Este, através da informação relativa recebida pelas diferentes câmaras possibilita a identificação e estimativa da localização de pessoas.

Segundos os autores, o sistema funciona bem até três pessoas permitindo a localização das pessoas a andar, paradas, sentadas ou mesmo próximas. Roupas de cor idêntica poderão levar a erros no processo de localização.

3.3 . Sistemas baseados em sinais *wireless*

Este tipo de sistemas é sem dúvida o mais implementado no mercado actual. Estes podem ser subdivididos consoante o tipo de sinal utilizado para a localização em: infravermelhos, ultra-sons, rádio frequência ou híbridos. Os sistemas baseados em rádio-frequência ainda podem ser divididos consoante a tecnologia que for utilizada para o processo de localização, entre elas as de maior destaque são: *RFID; UWB; Wi-Fi; Bluetooth e ZigBee*.

Neste sub capítulo apenas iremos descrever alguns desses sistemas descrevendo o seu modo típico de funcionamento.

3.3.1 *Active Badge [17]*

Este sistema foi desenvolvido no *Olivetti Research Laboratory* (agora *AT&T Cambridge*) e usa a tecnologia de difusão infravermelha para realizar a estimativa da localização de objectos.

É composto por crachás electrónicos que enviam periodicamente informações de identificação (o seu *ID*) para uma rede de sensores de referência previamente instalada na área de localização. Através de transmissores e receptores infravermelhos a comunicação entre os crachás e a rede é possível.

A localização é estimada através da identificação da posição dos sensores que detectam os sinais emitidos pelos crachás, sendo assim estimada a posição baseada em proximidade.

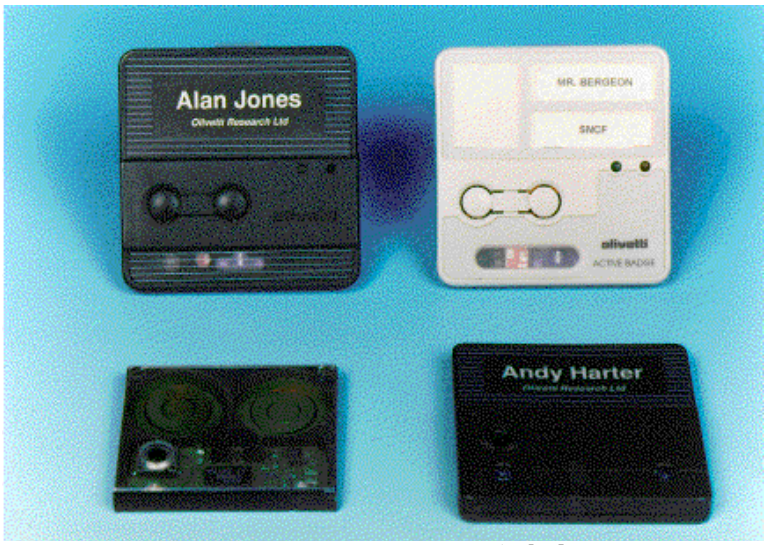

Imagem 3.6: Sensor Active Badge[35]

Este sistema tem alcance reduzido, cerca de 5 metros, sendo também limitado pelas posições dos sensores de referência. Revela baixo desempenho com presença de luz solar directa.

Os autores revelam que o alcance do sistema é limitado devido à área de cobertura reduzida dos sensores e da necessidade de *LOS*, necessitando de um grande esforço para distribuir os sensores de modo a cobrir toda a área de um ambiente *indoor*.

3.3.2 *Active Bat [21]*

Este sistema é baseado no *ToA* entre sinais ultra-sónicos e sinais de rádio-frequência.

O sistema é descrito pela Imagem 3.7. Consiste numa rede de sensores de referência (normalmente instalados no tecto) que comunicam com um controlador central que irá realizar o cálculo da localização do *Bat* (dispositivo móvel) através do processamento de vários tempos de chegada de diferentes receptores de referência.

O *Bat* é constituído por um microprocessador, um transceptor rádio de 418 *MHz*, um *FPGA*, e um vector hemisférico de 5 transdutores ultra-sónicos.

Neste sistema uma estação base de rádio frequência envia pedido (sinal *RF*) para a rede que será detectado tanto pelos *Bats* como pelos receptores de referência. O *Bat*, ao receber este sinal emite um sinal ultra-sónico para a rede de sensores de referência montada no tecto. Os sensores de referência, que também recebem o sinal *RF*, medem a diferença temporal entre este sinal e o sinal ultra-sónico emitido pelo *Bat*. Em seguida, esta informação será enviada para um controlador central que por sua vez irá estimar a localização do *Bat* por *lateration*.

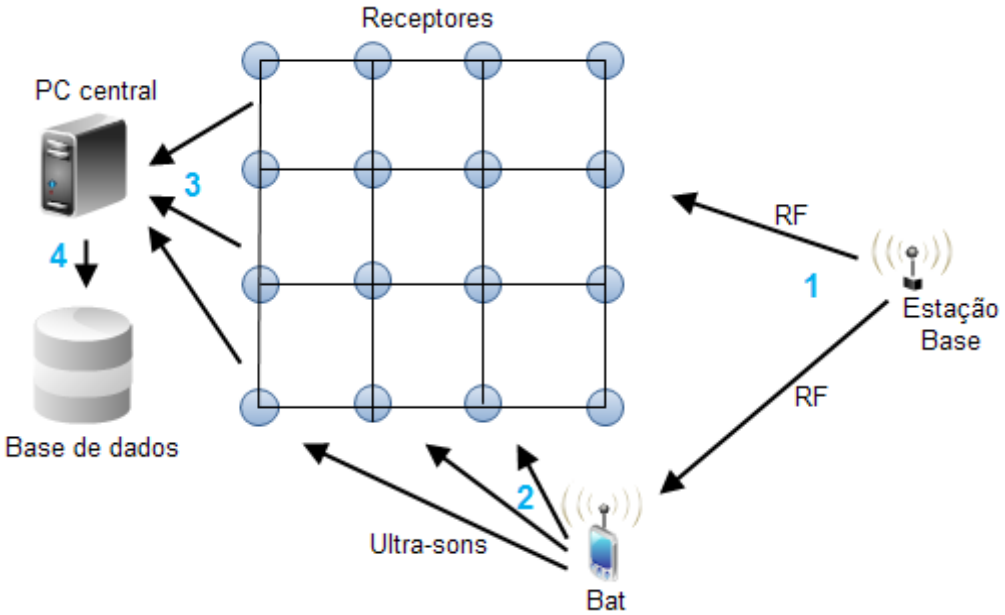

Imagem 3.7: Sistema Active Bat

Os autores de [21] referem um precisão de 95% dentro de 8cm quando são consideradas mais de 10 amostras.

3.3.3. *Cricket [11]*

Este sistema aplica a técnica *TDoA* baseada na diferença temporal entre vários sinais ultra-sónicos e sinais *RF* para estimar a localização do sensor móvel.

Existem dois tipos de dispositivos *Cricket* [11]: *beacons* que funcionam como nós de referência tipicamente no tecto ou paredes, e *listeners*, objectos móveis que calculam a própria localização.

Imagem 3.8: Módulo Cricket[11]

Os *beacons* enviam periodicamente mensagens *RF* informando a sua identificação *(ID)* e posição. No início de cada mensagem o *beacon* transmite também um pulso ultra-sónico.

Como a velocidade do sinal *RF* é muito superior que a velocidade do som, irá haver um desfasamento temporal entre os dois sinais. Quando os *listeners* recebem o sinal *RF* de um *beacon*, seguido do sinal ultra-sónico, estes medem o desfasamento entre os dois sinais.

Os *listeners* podem agora estimar a sua posição de dois modos: triangulação da posição quando recebem informação de diversos beacons ou através de técnicas de proximidade quando apenas recebem informações de um único *beacon*.

Este sistemas permite manter a privacidade dos *listeners* em relação à rede de sensores, mas por outro lado exigem maior complexidade destes pois têm que executar os algoritmos de estimativa da localização. Apresentam uma exactidão inferior a 30 cm em 99% dos casos [11].

3.3.4. *RADAR [18]*

Este é um sistema baseado em rádio-frequência que utiliza dispositivos com o padrão *IEEE 802.11*. Este sistema é formado por várias estações base (pontos de acesso) de modo a possibilitar uma maior cobertura do sinal. A estimativa da localização é realizada através de uma análise de cenário ou através de técnicas de triangulação [18].

Neste sistema é necessário fazer uma calibração do sistema (fase *offline*) criando assim um mapa com os valores de potências do sinal para cada posição do ambiente de localização. Na fase *online*, os valores da potências de sinal captados serão comparadas com este mapa estimando assim a localização do dispositivo.

RADAR calcula a posição aplicando o algoritmo *kNN*. Este algoritmo calcula a distância euclidiana entre o conjunto de sinais recebidos e o mapa de calibração. A localização que apresentar menor distância euclidiana será considerada como a posição do objecto localizado.

Este sistema implica um grande esforço para a fase de calibração e uma necessidade de actualização desta caso se verifiquem alteração do ambiente de localização.

Em [18] foram usados dois métodos, um método empírico e outro determinístico. O primeiro usa os dados empíricos recebidos na fase de calibração para construir o espaço de localização para o algoritmo *kNN*. No segundo, foram empregues modelos de propagação onde é estimado a distância dos sensores de referência aos sensores móveis através da relação entre o nível de potência recebido e o modelo utilizado. Através da informação das distâncias entre os dispositivos e sabendo previamente a posição dos sensores de referência é efectuado uma triangulação para estimar a localização. Foram obtidos melhores resultados através do método empírico, com um erro de 1.92, 2.94, 4.69 metros em 25, 50 e 75% das localizações [18].

3.3.5 *LANDMARC*[36]

O *LANDMARC (Location Identification based on Dynamic Active RFID Calibration)* é um sistema que usa a tecnologia *RFID* para estimar a localização de objectos.

Este sistema é constituído por leitores *RFID* que conseguem ler a informação emitida pelas *tags RFID*. Existem dois tipos de *tags*, activas e passivas. As *tags* passivas não necessitam de baterias, são baratas, tamanho reduzido, mas têm alcance limitado.

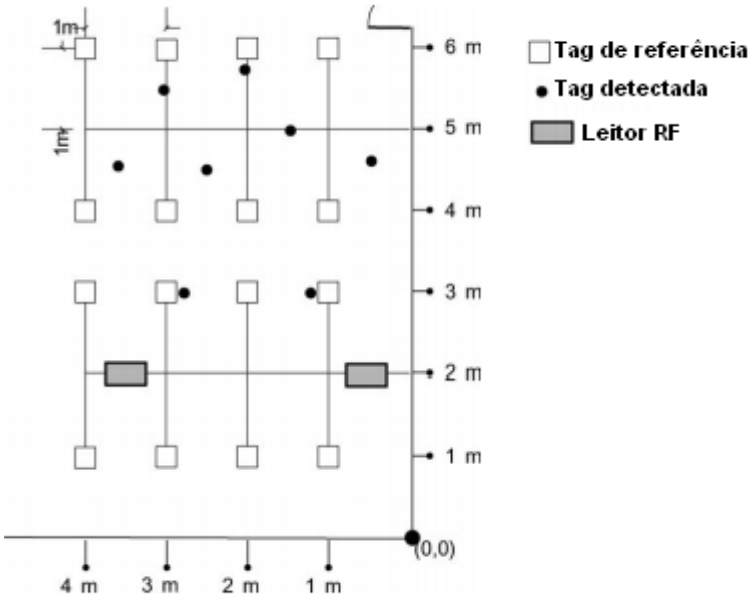

Imagem 3.9: Sistema LandMarc [36]

As *tags* activas, em relação às passivas, têm uma bateria incorporada e maior alcance. Mas por sua vez têm um custo mais elevado.

A infra-estrutura usada pelo *LANDMARC* é constituída por leitores *RFID*, *tags* activas *RFID* e um servidor que efectua a comunicação com os leitores e o cálculo da estimativa da localização.

Nesta infra-estrutura, são conhecidas previamente as coordenadas das *tags* de referência que possibilitam a base para o cálculo das *tags* que se pretendem localizar. Todas as *tags* emitem um sinal com o *ID* num ciclo médio de 7,5 em 7,5 segundos, tendo uma duração de vida média de 3-5 anos. Os leitores possibilitam 8 níveis diferentes de potência onde o nível 1 corresponde ao menor e 8 ao mais elevado.

O leitor mede a intensidade do sinal da *tag* localizável e a intensidade do sinal das *tags* de referência mais próximas. Em seguida envia essa informação para o servidor que através de um algoritmo de *kNN* estima a localização das *tags*.

A exactidão depende do número de *readers* utilizados e do número de *tags* de referência mais próximas da *tag* localizável.

Os autores revelam um erro máximo de 2 metros, e um erro inferior a 1 metro em 50% dos casos para uma disposição de uma *tag* de referência por metro quadrado [36].

3.3.6. *Ubisense [37]*

Esta tecnologia é baseada em *ultra-wide band (UWB)*. Foi desenvolvida pela universidade de *Cambridge* e teve como base a construção de um sistema de localização em tempo real *(RTLS)* que proporciona um elevado nível de exactidão relativamente aos sistemas utilizando *RFID* ou *Wi-Fi*, podendo ir até 15cm de exactidão.

Este sistema consiste em três componentes: *tags* activas que transmitem pulsos *UWB*; sensores de referência colocados numa infra-estrutura fixa que recebe os sinais emitidos pelas *tags* e uma plataforma de *software* que permite visualizar a posição.

As *tags* têm um transmissor *UWB* e um transceptor rádio 2.4*GHz* que permite a troca de mensagens de controlo bidireccional entre a *tag* e os sensores. Este controlo dinâmico permite preservar a duração das baterias e assegurar uma taxa de transmissão de mensagens adequada para as necessidades da aplicação.

As *tags* transmitem pulsos *UWB* a uma frequência de 6 a 8 *GHz* com uma duração bastante reduzida (na ordem de 1 nano segundo) quando um sensor de referência lhes faz um pedido, sendo a taxa de envio destes sinais controlada pelos sensores.

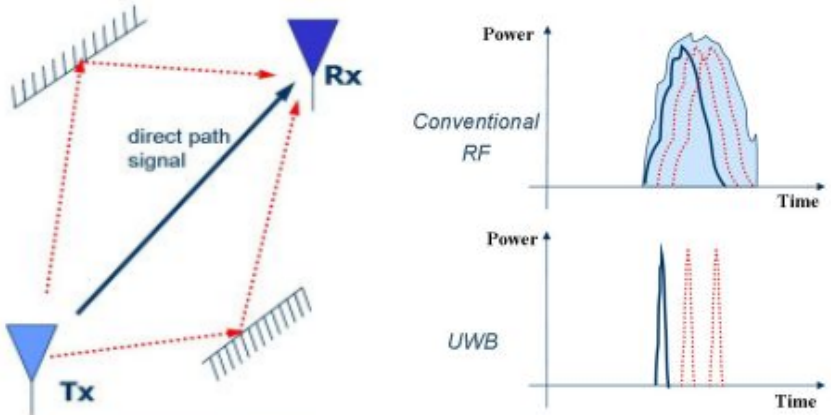

Imagem 3.10: Influência do efeito de multi-percursos em RF e UWB[37]

Os sensores são muito sensíveis aos pulsos *UWB* de baixa duração e potência (menor que 1*mW*) emitidos pelas *tags* possibilitando assim uma filtragem mais fácil e distinção entre sinais reflectidos de sinais em *LOS*, representado na Imagem 3.10.

Cada sensor de referência detecta o ângulo de chegada (*AoA*) do sinal emitido pela *tag* através de um *array* de antenas. Caso os sensores estejam sincronizados entre si, é possível associar o uso de *AoA* com *TDoA*. Segundo a literatura, com a associação das duas técnicas é possível uma exactidão até 15cm [37].

Os sensores comunicam e são programados via *Ethernet* ou via *wireless* dependendo da arquitectura do sistema implementado.

Todas estas informações dos sensores serão transmitidas para a plataforma de *software* que processará toda a informação permitindo assim a visualização da localização num ambiente gráfico, assim como várias informações que poderão estar associadas aos sensores.

3.3.7. *ZigBee Location Engine [38][6]*

A *Texas Instruments* desenvolveu o *chip CC2431*, que é um componente de arquitectura *System-On-Chip (SOC)* que permite a localização em redes *wireless* através do protocolo *ZigBee/IEEE 802.15.4.*

O *chip* inclui um módulo implementado em *hardware*, integrado no *chip*, que pode ser utilizado nos dispositivos terminais para calcular a posição do módulo que se pretende localizar designado por *location engine*.

Para realizar a localização são necessários três tipos de nós [39]:

Blind Node: nó com localização desconhecida que, ao receber um pedido de localização calcula a sua posição baseada no *location engine*. Estes nós têm que conter o l*ocation engine TI CC2431*. Posteriormente envia a sua localização para o *Location Dongle*.

Reference Node: um dispositivo estático cuja posição (X,Y) é conhecida, e que informa os outros dispositivos da sua localização sempre que receber um pedido para tal.

Location Dongle: dispositivo que possibilita a ligação entre as aplicações de um computador e a rede constituída por *Blind Nodes e Reference Nodes.* Permite assim a configuração da rede de sensores e a recepção das mensagens de localização.

De um modo simplificado, o processo de localização é efectuado através dos seguintes passos:

1. Um *Blind Node* envia uma mensagem *broadcast* a indicar o pedido de localização para todos os *Reference Nodes* ao seu alcance (apenas um salto);
2. Os *Reference Nodes* respondem para o *Blind* com uma mensagem a informar a sua posição (X,Y) e o valor de *RSSI* registado na recepção do pedido de localização enviada pelo *BlindNode:*
3. O *Blind Node* utiliza o *location engine* para calcular a posição por triangulação. Ao

receber a mensagem enviada por um *RefNode* a indicar a posição e valor de *RSSI* registado, o l*ocation engine* através de um modelo de perdas por propagação do sinal *RF* previamente calibrado calcula a distância a esse *RefNode*. Com um mínimo de três distâncias calculadas e com a informação da posição fixa dos *RefNodes* é possível estimar a posição do *BlindNode* por *laceration*. Em seguida envia uma mensagem com a posição calculada para o *Location Dongle*;

4. O *Location Dongle* reencaminha essa informação para uma aplicação, que por sua vez permite uma fácil visualização e identificação do *Blind* numa interface com o utilizador.

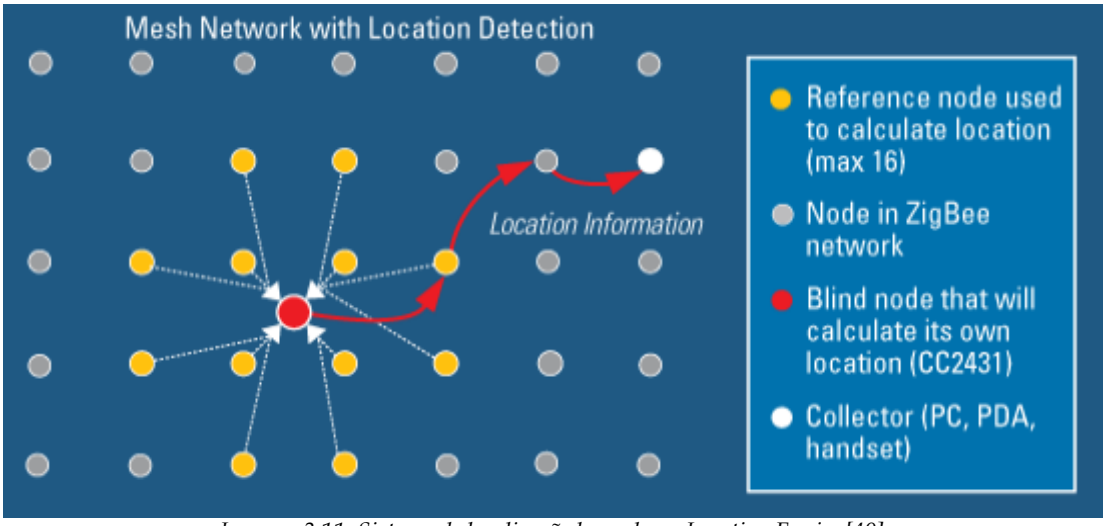

Imagem 3.11: Sistema de localização baseado no Location Engine[40]

A exactidão deste módulo para aplicações típicas é cerca de 3 a 5 metros [38]. Este módulo é recomendado para aplicações onde um baixo consumo energético é essencial. Possibilitando efectuar o processo de localização com intervalos prolongados de adormecimento do módulo.

3.4 . Comparação entre sistemas e tecnologias de localização

Na Tabela 3.1 são descritos vários sistemas de localização *indoor* assim como as suas características de modo a ser possível uma visão global sobre estes.

Sistema	Tecnologia	Algoritmo	Exactidão	Precisão	Custo
RADAR	*WLAN (RSSI)*	*kNN*	*3 a 5m*	*50% a 2.94 m, 75% a 4,69 metros*	*Baixo*
Active Bat	*ToA de Ultra-sons*	*Lateration*	*5 cm*	*95% dentro de 8cm*	*Médio Alto*
Horus	*WLAN (RSSI)*	*Método probabilístico*	*2 metros*	*90% em 2.1m*	*Baixo*
Ekahau	*WLAN (RSSI)*	*Método probabilístico*	*1 metro*	*50% em 2 metros*	*Baixo*
Ubisense	*Unidireccional UWB TDoA + AoA*	*Mínimo desvio quadrado*	*15cm*	*99% em 0,3m*	*Médio Alto*
LANDMARC	*Active RFID (RSSI)*	*kNN*	*<2m*	*50% em 1m*	*Baixo*
TOPAZ	*Bluetooth (RSSI) + IR*	*Proximidade*	*2m*	*95% em 2m*	*Médio*
Cricket	*TDoA de Ultra-sons*	*Proximidade , Lateration*	*15cm*	*99% dentro de 30cm*	*Médio Alto*
GSM fingerprinting	*GSM cellular network por RSSI*	*KNN com pesos*	*5m*	*80% em 10m*	*Médio*
SoC cc2431 Location Engine	*ZigBee (RSSI)*	*Lateration por RSSI*	*3-5m*	*~95% dentro de 5m*	*Baixo*

Tabela 3.1: Comparação entre sistemas de localização[7]

De um modo geral a tecnologia *RFID* passiva tem as suas grandes vantagens no preço das *tags* e no consumo energético visto não necessitarem de baterias, mas por outro lado têm alcance reduzido e um custo elevado dos *readers,* tornando-o pouco apropriado para uma localização residencial.

A tecnologia *UWB* apresenta uma resolução muito boa, baixo consumo energético devido ao baixo *duty cycle* mas os leitores ainda são relativamente caros. Como o alcance é reduzido, para uma localização residencial seriam necessários vários sensores o que torna dispendioso o sistema.

A tecnologia *Wi-Fi* tem a vantagem de estar implementada na maioria das residências provendo assim a infra-estrutura para sistemas de localização, mas tanto esta tecnologia como o *Bluetooth* não foram especialmente desenvolvidos para baixo consumo

energético. A tecnologia *ZigBee* foi especialmente desenvolvida para sistemas de baixo consumo que não necessitam de elevada taxa de transmissão, ideal para redes de sensores. Devido à sua simplicidade de *hardware* e pilha protocolar simples é possível criar dispositivos com preço bastante atractivo. Isto faz desta tecnologia apropriada para sistemas de localização de baixo custo, elevada durabilidade mas que não necessitem de uma exactidão elevada.

RFID e UWB poderão ser tecnologias ascendentes nesta área caso o preço dos *readers* baixe para preços mais competitivos. Neste momento, sistemas baseados em *SoC (system on chip)* de baixo consumo revelam-se bastante apropriados para a área de localização.

Na Tabela 3.2 é descrito um resumo comparativo entre diferentes tecnologias de localização *indoor*. Esta tabela apenas relaciona os parâmetros genéricos de cada tecnologia tendo como objectivo uma fácil e imediata comparação entre tecnologias e não uma comparação pormenorizada e detalhada. Conforme os sistemas considerados, alguns parâmetros poderão obter diferentes valores. A avaliação é descrita de 1 a 5, onde valores superiores correspondem a uma tecnologia mais adequada ao parâmetro descrito.

Tecnologia	***Consumo da Tag***	***Resolução***	***Alcance***	***Custo da Tag***	***Custo do reader***
RFID passivo	5	1	1	5	1
Bluetooth	2	3	3	2	3
UWB	4	4	2	4	2
Wi-Fi	1	3	5	2	3
ZigBee	4	3	4	3	4

Tabela 3.2: Comparação entre diferentes tecnologias de localização

Capítulo 4

ZigBee

Neste capítulo é descrita a tecnologia *ZigBee* utilizada no sistema implementado. Na secção 4.1 . é descrita uma breve referência histórica sobre a tecnologia. Em 4.2 . são descritas as principais características da tecnologia *ZigBee* e em 4.3 . é descrito a sua arquitectura protocolar. Nas secções seguintes são descritas com maior pormenor as quatro camadas que constituem a pilha protocolar, camada física, controlo de acesso ao meio, rede e aplicação.

4.1 . Referências históricas

Uma forte área em expansão hoje em dia diz respeito a sistemas do controlo e monitorização de sensores. Estes sistemas geralmente não necessitam taxas de transmissão superiores a dezenas de *bytes*. Até há relativamente poucos anos não havia nenhum padrão sem fios globalmente aceite para este tipo de aplicações que o mercado começava a exigir. Face a esta necessidade a organização *IEEE* desenvolveu a base

necessária para este tipo de tecnologia lançando em Maio de 2003 o padrão *IEEE 802.15.4*. Este padrão robusto para redes sem fios de curtos alcance, permitiu a identificação única dos transceptores de rádio dentro de uma rede assim como definiu o modo e formato de comunicação entre os transceptores. Especificaram assim as camada física *(PHY – Physical Layer)* e de controlo de acesso ao meio *(MAC - Media Access Control),* fornecendo a base protocolar para redes de sensores.

Uma aliança de empresas de diferentes segmentos designada "*ZigBee Alliance*" [41] fez esforços para proporcionar e desenvolver a tecnologia e a camada protocolar acima do *IEEE 802.15.4*.

Surgiu assim, em Dezembro de 2004, as especificações do padrão *ZigBee 802.15.4*.

Hoje em dia a *ZigBee Alliance* é constituída por mais de 300 empresas das mais diferentes áreas como por exemplo, *Freescale, Texas instrumentos, Samsung, Philips, Schneider Electric, Siemens, Microchip, Motorola, Mitsubishi Electric[41].*

O *ZigBee* teve como objectivos iniciais a transmissão sem fios, o consumo energético reduzido, a alta fiabilidade e potencializar um protocolo global livre. Esta última razão levou a que em Junho de 2005 a *ZigBee Alliance* lançasse as suas especificações ao público de modo a permitir aos produtores uniformizar o desenvolvimento de aplicações e dispositivos nesta área. Assim, criou as condições para o abandono de sistemas proprietários distintos e dar lugar a um *standard*, que ao ser compatível, proporcionaria evidentes vantagens de interoperabilidade. Segundo a própria definição da *ZigBee Alliance*, o seu objectivo é "*To enable reliable, cost-effective, low-power, wirelessly networked, monitoring and control products based on an open global standard*" [41].

Posteriormente já foram lançadas mais duas versões de especificações, em Setembro de 2006 foi lançado o *ZigBee 2006 Specification* e em finais de 2007 foi lançado o *ZigBee Pro*.

O nome *ZigBee* surge da analogia entre o funcionamento de uma das suas topologias, topologia em malha *(mesh)*, e do modo como as abelhas se deslocam. Na topologia em malha a transferência de informação entre dispositivos pode seguir várias rotas, dando a ideia de um ziguezague da informação. Do mesmo modo as abelhas deslocam-se em ziguezague e desta forma trocam informações com outros elementos da colmeia como por exemplo: distância, direcção e localização do local onde se encontram os alimentos.

4.2 . Características da tecnologia *ZigBee*

Muitas são as razões que fazem desta tecnologia diferente de todas as outras, as suas principais características são:

- consumo energético reduzido, essencial para a área de monitorização;
- pilha protocolar simples;
- admissão de diferentes topologias de rede: estrela, malha ou árvore;
- possibilidade de suportar um número quase ilimitado de nós por rede (máximo de 65535 dispositivos por cada coordenador *ZigBee*);
- apresenta uma baixa latência;
- reduzido tempo de ligação à rede;
- rápida transição na passagem do modo *standby* a activo;
- permite dois modos de operação: activo e modo *sleep*;
- dois modos de operação da rede: *beacon* habilitado e *beacon* não habilitado.
- elevada segurança, com recurso a *128-bit encryption*;
- elevada fiabilidade;
- suporte para duas classes de dispositivos físicos *(FFD - Full function device ou RFD - reduced function device)* divididos em 3 tipos lógicos (coordenador, roteador e terminal).

4.3 . Arquitectura Protocolar

A pilha protocolar *ZigBee* é baseada no modelo *OSI (Open Systems Interconnections) e* composta por quatro camadas: física *(PHY),* controlo de acesso ao meio *(MAC)*, rede *(NWK)* e aplicação *(AP)*.

A definição e desenvolvimento da camada física e da camada de controlo de acesso ao meio foram realizadas pelo grupo de trabalho *802.15.4,* sendo as restantes camadas, camada de rede e aplicação definidas pela *aliança ZigBee*.

Cada camada executa serviços específicos que podem ser acedidos pela camada superior, serviço de transmissão e serviço de gestão.

Existem dois tipos de entidades, a entidade de dados que fornece dados para o serviço de transmissão e a entidade de gestão que fornece informação para todos os outros serviços. Cada entidade de serviço expõe uma interface para a camada superior

através do ponto de acesso ao serviço *(SAP – Service Access Point)* e cada *SAP* suporta um número de primitivas de serviço para activar a funcionalidade que se pretende solicitar.

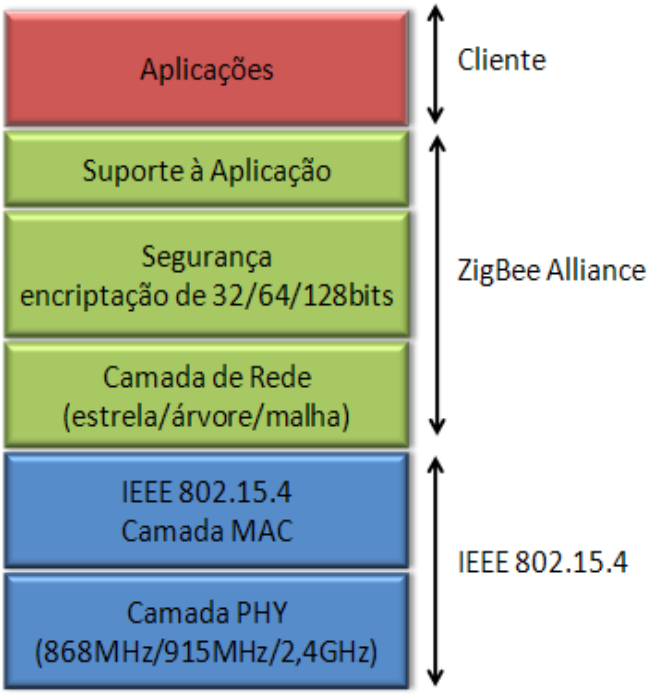

Imagem 4.1: Pilha protocolar simplificada

A camada física é responsável pela transmissão e recepção de mensagens através do canal físico *RF*. Das suas funções fazem parte a activação e desactivação do *transceiver*, detecção da energia *(ED – Receiver Energy Detection)*, indicação da qualidade da ligação *(LQI – Link Quality Indication)*, selecção do canal e transmissão/recepção de tramas através do meio físico.

A camada *MAC* é responsável por controlar o acesso aos canais *RF*, através do uso de mecanismos de prevenção de colisão *CSMA-CA (Carrier Sense Multiple Access – Collision Avoidance)*. Nesta camada são definidos dois dispositivos consoante as funcionalidades (*FFD* ou *RFD*). Cabe também a esta camada a responsabilidade de garantir a sincronização e transmissão de *beacons* entre os dispositivos, permitindo a fiabilidades de operação.

A camada de rede é responsável pelos mecanismos de associação/dissociação de elementos na rede, mecanismo de encaminhamento e manutenção de rotas entre os elementos da rede.

A camada de aplicação pretende assegurar uma correcta gestão e suporte para as diversas aplicações sendo dividida em três grandes blocos, sub-camada de suporte à aplicação (*APS – Application Support Sub-Layer),* objectos de aplicação *ZigBee* (*ZDO - ZigBee Device Objects)* e *FrameWork* da aplicação (*AF - Application FrameWork)*.

ZDO é responsável por definir a tarefa do dispositivo na rede (coordenador, roteador ou terminal – *end device*), por descobrir novos dispositivos na rede e de identificar os seus serviços. Pode posteriormente estabelecer uma rota segura e inicializar ou responder a pedidos de ligação (*bindings*).

APS funciona como a ligação entre a camada de rede e os outros componentes da camada de aplicação, o que oferece uma interface para serviços de controlo. Ela mantém as tabelas de ligação (*binding tables*) que permite ligar os dispositivos de acordo com os seus serviços e necessidades.

AF é o ambiente onde os objectos de aplicação *ZigBee* se encontram armazenados.

4.4 . Camada Física

4.4.1. Detalhes técnicos

A camada física proporciona a ligação entre a sub-camada *MAC* e o canal de rádio, através de *firmware e hardware RF*. A camada física contém uma entidade de gestão designada por *PLME* (*Physical Layer Management Entity*) e uma entidade de dados *PD (PHY Data)*. Estas entidades permitem a geração de primitivas que possibilitam a comunicação com as entidades da camada superior através dos *SAP, PD-SAP* e *PLME-SAP* correspondentes. A *PLME* também gere a base de informação da camada física, designada por *physical information base (PIB)*.

Estas duas entidades estão directamente relacionadas com dois tipos de serviços: serviço de dados *PHY* responsável pela transmissão e recepção de *PHY protocol data units (PPDU)* através do canal de rádio; e serviço de gestão *PHY*, responsável pela gerência dos vários elementos de *PHY*.

Para a camada física, estão definidas duas plataformas de *hardware* para o padrão *IEEE 802.15.4*. Uma descreve o espectro das bandas de frequência *ISM (industrial, scientific and medical)* 868*MHz* utilizada na Europa e 915*MHz* utilizada nos *EUA*; e outra

para a banda *ISM* 2,4*GHz* utilizada em todo o mundo.

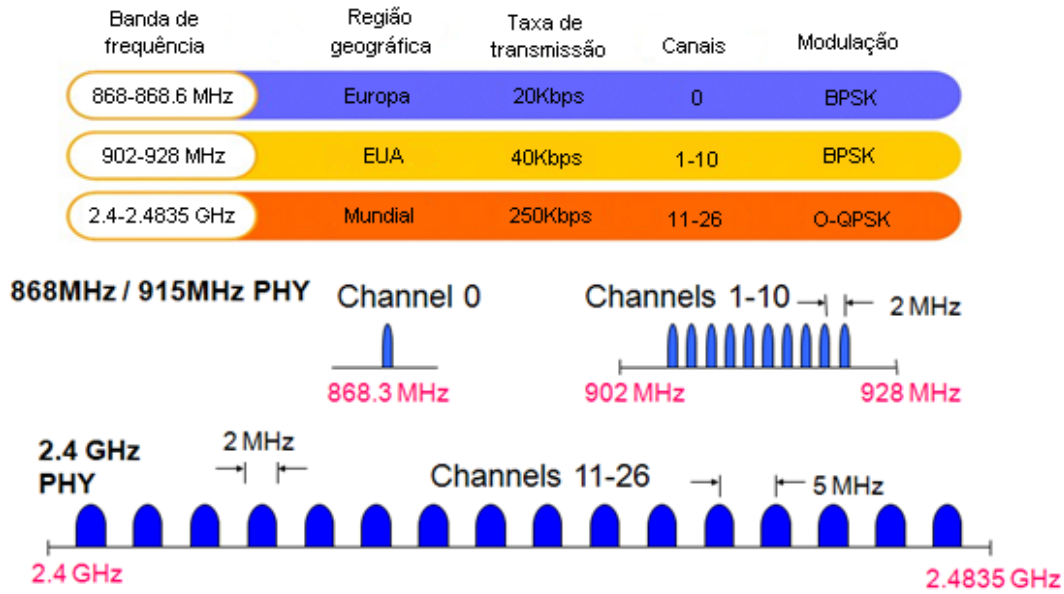

Imagem 4.2: Canais de transmissão usados na camada física e suas especificações [42]

Como referido na Imagem 4.2, existem 27 canais alocados pelo *IEEE 802.15.4* com três taxas de transmissão diferentes. Um canal opera entre 868 e 868.8*MHz* com taxa de transmissão de 20*Kbps*, 10 canais operam entre 902 e 928*MHz* a 40*Kbps* e 16 entre 2.4 e 2.4835G*Hz* a 250*Kbps*.

A modulação do padrão *IEEE 802.15.4* baseia-se em duas variantes da modulação *PSK (Phase-Shift Keying)*, ou seja, irá haver uma mudança de fase da portadora de acordo com a informação transmitida. Para as frequências 868/915 *MHz*, é usado a variante *BPSK (Binary Phase-Shift Keying)* e para a frequência de 2.4*GHz* é usada a modulação *O-QPSK (Offset – Quadrature Phase-Shift Keying)*.

4.4.2. Responsabilidade da camada física

Esta camada é responsável pelas seguintes funções:

- activação e desactivação do transceptor;
- transmissão e recepção de dados;
- selecção do canal de frequência;
- detecção de energia no canal (*ED*);
- indicação da qualidade do link (*LQI*);
- executar *CCA(clear channel assessment)* para o protocolo *CSMA-CA*.

- **Activação e desactivação do transceptor**

O transceptor tem 3 estados possíveis: transmissão, recepção ou adormecido. Através de um pedido enviado pela camada *MAC*, o transceptor pode ser ligado ou desligado. Esta é a grande característica que permite às tecnologias baseadas no padrão *IEEE 802.15.4* poupar energia.

- **Transmissão e recepção de dados**

A camada física é responsável pela transmissão e recepção de dados. A sensibilidade do receptor é de -85dBm para 2.4*GHz* e -94dBm for 868/915*MHz*.

- **Selecção do canal de frequência**

O *IEEE 802.15.4* permite o uso de 27 canais de frequência diferentes assim como a alteração do canal utilizado. Este pode ser útil para reduzir a interferência com outras tecnologias como por exemplo *Wi-Fi*, onde para evitar a interferência deveríamos escolher diferentes canais conforme a utilização na Europa ou EUA.

- **Detecção de energia no canal (*ED*)**

O transceptor é capaz de estimar o valor de potência recebida de um dado canal dentro da largura de banda do canal seleccionado. Este valor é fornecido num campo de um *byte*, desde 0x00 a 0xFF. O valor mínimo deve indicar uma potência recebida menor do que 10dB a cima da sensibilidade do receptor e a gama de potência recebida deve ter um mínimo de 40dB.

- ***Indicador da qualidade do link (LQI)***

O *LQI* caracteriza a intensidade e/ou qualidade da trama recebida. Este valor pode ser calculado apenas através da medida de *ED*, uma estimativa da relação sinal/ruído ou uma combinação desses dois. O valor de *LQI* é representado por um *byte*, sendo os valores máximo e mínimo associados aos valores de menor e maior qualidade dos sinais *IEEE 802.15.4* detectáveis pelo receptor. Todos os outros valores restringem-se a essa gama abrangida.

. *CCA* - clear channel assessment

CCA permite ao transceptor detectar se o meio deve ou não ser utilizado para transmissão. Sem este algoritmo seria muito difícil realizar comunicações sem fios, visto que diferentes sensores poderiam emitir ao mesmo tempo levando a colisões entre tramas. Esta detecção pode ser feita de acordo com 3 métodos distintos.

- Detecção de energia acima do nível do limiar de *ED*, onde *CCA* informará que o canal está ocupado sempre que detectar um valor superior ao nível limiar de *ED*.
- Detecção da portadora, onde *CCA* informará que o canal está ocupado sempre que detectar o sinal da portadora.
- Detecção da portadora acima do limiar de *ED*, onde *CCA* informará que o canal está ocupado sempre que detectar a portadora com energia superior ao nível do limiar de *ED*.

4.4.3. Estrutura da trama *PPDU*

A estrutura da trama *PPDU (PHY Protocol Data Unit)* é constituído por:

- Um cabeçalho de sincronização, *SHR (Synchronization Header)*, que permite a sincronização do receptor com o feixe de bits. Este contém 32bits no campo *Preamble* e 8 bits no campo *SFD (Start of Frame Delimiter)* que indica o fim de *SHR* e o início do campo de dados;
- Um cabeçalho da camada física, *PHR (PHY Header)*. Contém um campo de 7 bits *(FL – Frame Length)* que indica o comprimento em bytes do campo de dados *PSDU* (máximo de 127 *bytes*, representado por 111 1111) e um bit reservado.
- Campo de dados, *PSDU (PHY Service Data Unit)*, corresponde a um campo variável de tamanho máximo de 127 *bytes*, designado por *PHY payload*.

<table>
<tr><td colspan="2"></td><td colspan="3">Octets</td></tr>
<tr><td colspan="2"></td><td colspan="2">1</td><td>variable</td></tr>
<tr><td>Preamble</td><td>SFD</td><td>Frame length
(7 bits)</td><td>Reserved
(1 bit)</td><td>PSDU</td></tr>
<tr><td colspan="2">SHR</td><td colspan="2">PHR</td><td>PHY payload</td></tr>
</table>

Imagem 4.3: Descrição da trama PPDU [43]

4.5 . Camada de Acesso ao Meio (*MAC*)

É nesta camada que estão definidos os dois tipos de dispositivos da norma *IEEE 802.15.4*, sendo eles *FFD (Full Function Device) e RFD (Reduced Function Device)*. O *FFD* é capaz de implementar toda a *Z-Stack* podendo assim operar em qualquer um dos três níveis lógicos (coordenador, roteador ou *end device*), este pode comunicar com qualquer *FFD ou RFD*, sendo um dispositivo de maior complexidade. O *RFD* opera apenas como *end device* sendo utilizado para funções mais simples. Este só pode comunicar com *FFD*.

Dispositivo ZigBee	***Dispositivo IEEE 802.15.4***
Coordenador (coordinator)	*FFD*
Roteador (router)	*FFD*
Terminal (End Device)	*FFD ou RFD*

Tabela 4.1: Tipos de dispositivos

A camada de controlo de acesso ao meio é responsável pelos mecanismos de:

- gestão e sincronização de *beacons* (apenas gerados pelo coordenador);
- suporte de associação/dissociação de nós na rede de sensores;
- validação e reconhecimento de mensagens;
- gestão do acesso ao canal via *CSMA-CA*;
- manutenção dos *slots* de tempo reservados *(GTS)*.

4.5.1. Modos de comunicação

A camada de controlo de acesso ao meio suporta dois modos de comunicação, *beacon habilitado e beacon não habilitado (non beacon)*, sendo o *beacon* uma trama de controlo que limita temporalmente e regula a comunicação entre o coordenador e os outros dispositivos permitindo assim a sua sincronização. É garantido assim, uma baixa latência e fiabilidade nas comunicações.

Modo non beacon - após a fase de associação à rede, os dispositivos concorrem entre si através de mecanismo de *CSMA-CA* para aceder ao canal rádio.

Modo beacon – é enviado periodicamente *beacons* para os elementos da rede permitindo a sincronização e a gestão da transmissão/recepção de tramas. Ao tempo entre dois *beacons* consecutivos designa-se por *superframe*. No primeiro intervalo de tempo de cada *superframe* é enviado o *beacon*.

4.5.2. Mecanismo *CSMA-CA*

CSMA-CA (Carrier Sense Multiple Access with Collision Avoidance) é o mecanismo responsável pelo acesso ao canal de cada dispositivo.

Cada dispositivo, sempre que necessita de transmitir uma trama de dados ou comandos *MAC*, espera um período aleatório de tempo. Se após esta espera o canal for encontrado livre, ele transmite a trama. Caso ele encontre o canal ocupado irá esperar uma tempo aleatório maior, de modo a reduzir a probabilidade de colisão, e tentará de novo aceder ao canal.

O uso da comunicação *beacon* tem a vantagem de permitir acesso temporal garantido *(GTS)* a dispositivos, esta informação está contida no *beacon*. Assim, é possível garantir o acesso exclusivo do canal aos dispositivos pretendidos.

As tramas de reconhecimento são enviadas sem usar *CSMA-CA*.

4.5.3. Superframe

Ao tempo entre dois *beacons* consecutivos designa-se por *superframe*. No primeiro intervalo de tempo de cada *superframe* é enviado o *beacon*, este é usado para sincronização de dispositivos associados, identificar a *PAN (Personal Area Network)* e descrever a estrutura dos *superframes*.

Podemos dividir o *superframe* em dois períodos distintos, período activo durante o qual se processa a comunicação e período inactivo (opcional), durante o qual os dispositivos podem desligar os transceptores de modo a conservar energia.

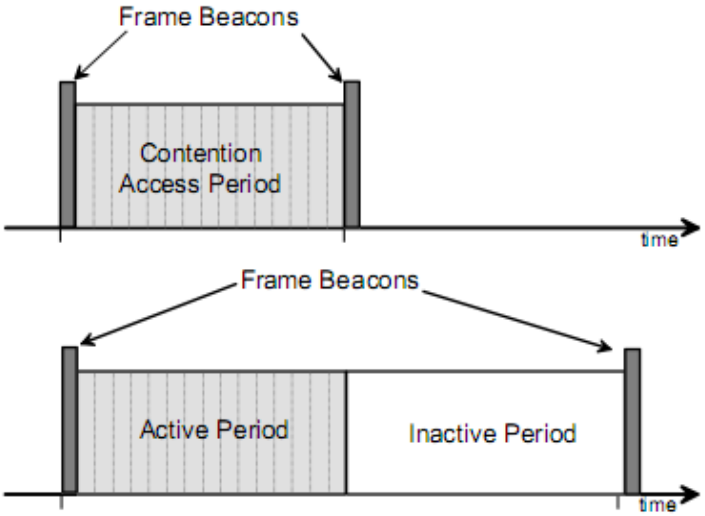

Imagem 4.4: SuperFrame sem e com período inactivo[43]

O período activo é dividido em 16 divisões temporais iguais *(slots)*. Seguido do *beacon* vem o *período de acesso de contenção (CAP – contention access period)*. Este período é destinado à comunicação dos dispositivos através do mecanismo *sloted CSMA-CA*, isto porque os períodos comunicação têm que ser alinhados com os limites dos períodos temporais.

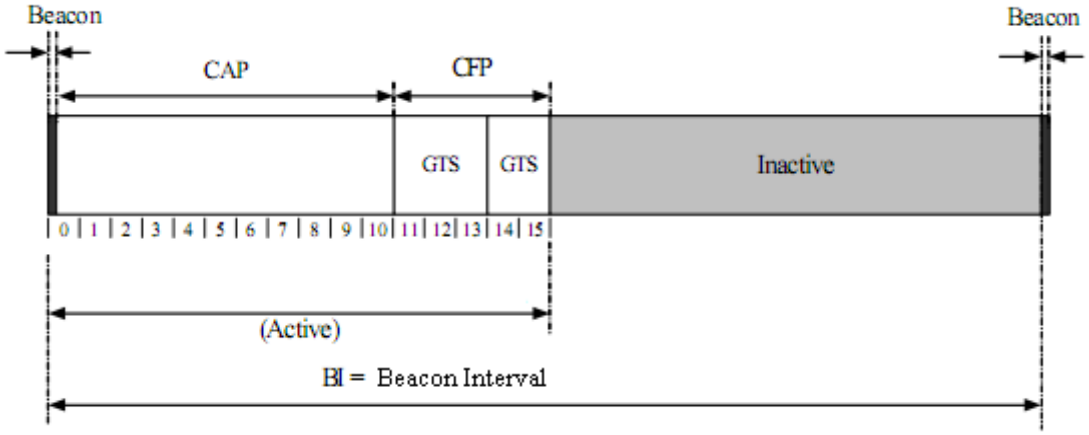

Imagem 4.5: Estrutura de um SuperFrame[43]

O *CAP* deve conter um mínimo de 8 períodos temporais mas pode ser alargado até 16. Seguido do *CAP* podemos ter um um *período livre de contenção (CFP – contention free period)*, sendo este opcional que poderá ir até um máximo de 7 períodos temporais, representando o final do período activo do *superframe*.

O coordenador pode reservar períodos temporais, obrigatoriamente múltiplos do mínimo período temporal, para a comunicação de dispositivos seleccionados. Estas reservas temporais designam-se por *GTS (guaranteed time slot)* e devem ser contíguas. Permitem um acesso exclusivo do canal para um determinado dispositivo, não executando o mecanismo de *CSMA-CA*. Esse dispositivo apenas pode transmitir ou receber dados, mas nunca ambos. As especificações do intervalo *GTS*, comprimento da mensagem e endereço dos dispositivos associados são enviadas no *beacon*.

4.5.4. Estrutura da trama *MPDU*

O formato da *MAC Protocol Data Unit (MPDU)* é constituído por três campos, *MAC Header, MAC Payout*, também designado por *MAC Service Data Unit (MSDU)* e pelo *MAC footer (MFR)*.

O campo *MHR* consiste no *frame control* que especifica o tipo da trama e nos campos

sequence Number e addressing Fields que permitem a identificação da mensagem e do endereço do módulo destino. Pode ainda ser adicionado um campo auxiliar para segurança, *Auxiliary Security Header*. O campo *Mac Sevice Data Unit (MSDU)* define o campo de dados usado pela camada de mais alto nível. Por fim, o *MAC footer* é composto pelo *Frame Check Sequence (FCS)* que valida a integridade da trama.

<table>
<tr><td>Octets: 2</td><td>1</td><td>0/2</td><td>0/2/8</td><td>0/2</td><td>0/2/8</td><td>0/5/6/10/ 14</td><td>variable</td><td>2</td></tr>
<tr><td rowspan="2">Frame Control</td><td rowspan="2">Sequence Number</td><td>Destination PAN Identifier</td><td>Destination Address</td><td>Source PAN Identifier</td><td>Source Address</td><td rowspan="2">Auxiliary Security Header</td><td rowspan="2">Frame Payload</td><td rowspan="2">FCS</td></tr>
<tr><td colspan="5">Addressing fields</td></tr>
<tr><td colspan="7">MHR</td><td>MAC Payload</td><td>MFR</td></tr>
</table>

Imagem 4.6: Formato de um trama típica MPDU[43]

As tramas da camada *MAC* podem ser divididas em:

- ***Trama de comando (MAC command)*** – transferência de comandos de controlo;
- ***Trama de dado*** – transferência de dados, máximo de 104 *bytes*;
- ***Trama de reconhecimento (ACK)*** – confirmação de recepção de uma trama;
- ***Tramas de beacon*** – transmissão de *beacons* (efectuada pelo coordenador).

4.5.5. Transferência de dados

Existem 3 tipos de transferência de dados possíveis: do coordenador para um dispositivo; de um dispositivo para o coordenador e entre dois dispositivos ponto a ponto. Na topologia em estrela apenas são efectuados os dois primeiros tipos de transferência de dados, na topologia de árvore ou malha as 3 são possíveis. Estes 3 tipos de transferência serão diferentes conforme o modo de comunicação, *beacon ou non beacon*.

- **Coordenador para um dispositivo**

Modo beacon habilitado (beacon-enabled)

O coordenador indica no *beacon* da rede que a mensagem de dados está pendente. O dispositivo escuta o *beacon* e envia um comando *MAC* requisitando o envio de dados usando *slotted CSMA-CA*. O coordenador recebe esse pedido e retransmite-lhe uma trama de reconhecimento. Em seguida é enviado a trama pendente de dados via *slotted CSMA-CA*. Ao receber a trama de dados, o dispositivo reenvia uma trama de reconhecimento

opcional ao coordenador, que por sua vez irá remover a mensagem da lista de mensagens pendentes do *beacon*.

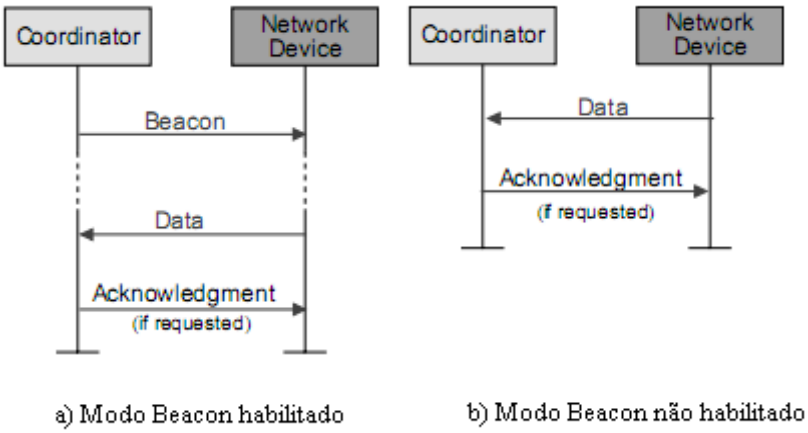

Imagem 4.7: Transferência de dados - coordenador para outro dispositivo[43]

Modo beacon não habilitado (non beacon-enabled)

O coordenador armazena os dados e espera por uma requisição de dados por parte do dispositivo. Este *MAC command* é enviado por *unslotted CSMA-CA* para o coordenador. Quando essa requisição é recebida pelo coordenador este transmite uma trama de reconhecimento.

Se houver dados pendentes, o coordenador reenvia-os para o dispositivo através de *unslotted CSMA-CA*. Caso não haja dados pendentes, o coordenador transmite um trama de dados com *payload* nulo indicando a não existência de dados pendentes.

- **Dispositivo para coordenador**

Modo beacon habilitado

O dispositivo aguarda a recepção do *beacon*, sincronizando-se com a estrutura *superframe*. No *CAP* este transmite a sua trama de dados para o coordenador, usando *slotted CSMA-CA*. Posteriormente o coordenador reconhece a recepção de dados e transmite um reconhecimento opcional.

Modo beacon não habilitado

O dispositivo transmite a sua trama de dados para o coordenador usando *unslotted CSMA-CA* quando verificar o canal livre. Em seguida recebe um reconhecimento opcional por parte do coordenador.

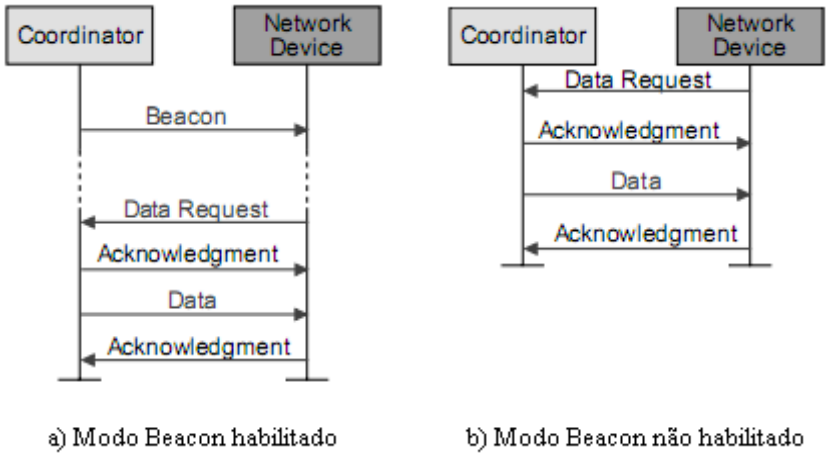

Imagem 4.8: Transferência de dados – dispositivo para o coordenador[43]

4.5.6. Associação e dissociação

A associação pode ser efectuada de um modo passivo, onde o dispositivo fica a escuta da transmissão de *beacons* por parte do coordenador, ou activa quando transmite ele próprio um pedido de *beacon*.

O procedimento de associação é iniciado através do envio de um pedido de associação para o coordenador de uma rede *PAN* já existente. Se este pedido for recebido pelo coordenador, este enviará uma trama de reconhecimento (não significa a aceitação de associação). O coordenador irá verificar se os recursos da redes permitem a associação do dispositivo, este processo é realizado num intervalo de tempo máximo de *aResponseWaitTime*. Caso se verifique a condição, o coordenador reservará um endereço curto para o dispositivo e enviará uma resposta de associação ao dispositivo indicando o seu endereço e o sucesso da associação. Caso não se verifique a condição, o coordenador enviará uma resposta indicando falha da associação. Em caso de sucesso da associação o dispositivo armazenará o endereço do coordenador.

Se o coordenador pretender dissociar um elemento envia um comando de notificação de dissociação para o dispositivo. Mesmo que uma resposta por parte do dispositivo não se verifique, o coordenador considerará o dispositivo como removido.

Se um dispositivo associado quiser deixar a rede, enviará uma notificação de dissociação ao coordenador. Como no caso anterior, o dispositivo irá dissociar-se mesmo que não receba reconhecimento por parte do coordenador. Todas as referências na rede do dispositivo serão removidas.

4.6 . Camada de Rede

A camada de rede é a primeira camada definida pela norma *ZigBee*. Nesta camada são implementadas as topologias de rede. A norma *ZigBee* e o padrão *IEEE 802.15.4* implementam as mesmas topologias, apenas diferem no tipo de dispositivos que constituem a rede. Estas topologias são: estrela *(Star),* árvore *(Cluster Tree)* e malha *(Mesh).*

Se considerarmos apenas *FFD e RFD* estaremos a referir-nos à topologia da camada *MAC*, se nos referirmos a coordenador, roteador e terminal estaremos a referir-nos à topologia da camada de rede.

Esta camada de rede é necessária para fornecer funcionalidades que garantam o correcto funcionamento da camada *MAC* do *IEEE 802.15.4* e para fornecer um serviço adequado para fazer a interface com a camada de aplicação. A interacção com a camada de aplicação é feita através de dois serviços, serviço de dados e serviço de gestão. O serviço de gestão contém um entidade *(NLME – Network Layer Management Entity)* que através do seu *SAP* fornecem os serviços necessários. Os serviços próprios do *NLME* são:

criação de uma rede – estabelecimento de uma rede;

associação ou dissociação na rede – capacidade de associar-se ou dissociar-se a uma a uma rede assim como a capacidade de requerer a dissociação de um dispositivo;

endereçamento – capacidade dos coordenadores e roteadores atribuírem endereços a dispositivos que se associem à rede;

descoberta da vizinhança – descoberta, registo e comunicação de informações relativas aos dispositivos vizinhos;

descoberta de rotas – descoberta e registo de rotas através da rede segundo o qual as mensagens podem ser devidamente redireccionadas;

No que diz respeito ao serviço de dados a camada de rede contém um entidade *(NLDE – Network Layer Data Entity)* que fornece o serviço de transmissão de dados através do *SAP*. Este possibilita os seguintes serviços:

Criação de *NPDUs (Network Protocol Data Units)* – o *NLDE* consegue gerar uma *NPDU* a partir de uma *PDU* da camada de Aplicação através da adição de um cabeçalho próprio da camada de rede;

Topologia específica de *routing* – o *NLDE é* capaz de transmitir uma *NPDU* para o devido dispositivo quer seja o destinatário final da comunicação ou o passo seguinte para chegar ao destinatário final;

Segurança – é a capacidade de garantir a autenticidade e confidencialidade de uma transmissão.

A camada de rede tem diferentes funcionalidades conforme o tipo de dispositivo a que nos tivermos a referir, estas diferenças estão descritas na Tabela 4.2.

Função	*Coordenador*	*Roteador*	*End Device*
Estabelecimento de uma nova rede	Sim	Não	Não
Atribuição de endereços de rede (2 bytes)	Sim	Sim	Não
Permissão para associação e dissociação de elementos na rede	Sim	Sim	Não
Manutenção de uma lista dos dispositivos vizinhos	Sim	Sim	Não
Reencaminhamento de tramas	Sim	Sim	Não
Transferência de tramas	Sim	Sim	Sim
Associação/dissociação de uma rede	Sim	Sim	Sim

Tabela 4.2: Diferentes funções dos elementos de uma rede ZigBee

4.6.1. Topologias de Rede

As redes com o padrão *IEEE 802.15.4 e ZigBee* podem operar em 3 tipos de topologia: topologia em estrela, árvore ou em malha. Na camada *MAC* os dispositivos são divididos consoantes a complexidade das suas funcionalidades em *FFD e RFD*; para a camada de rede são divididos em dispositivos lógicos consoante a tarefa que desempenham na rede. A relação entre eles é descrita na Tabela 4.1.

A topologia em estrela *(Star)* é caracterizada por uma comunicação onde todos os dispositivos estão ligados ao coordenador da *PAN*, não estabelecendo nenhum outro tipo de comunicação entre dispositivos. Esta topologia é indicada para redes centralizadas e para aplicações onde o tempo é essencial. Tem baixa fiabilidade pois não existem caminhos alternativos em caso de falha do parente (apenas coordenador).

A topologia em árvore é um pouco mais complexa do que a topologia em estrela pois permite um roteamento da comunicação apesar de cada elemento apenas ter uma única rota para comunicar com qualquer outro dispositivo. Podemos fazer a analogia do

coordenador ao tronco de uma árvore, sendo os *end devices* as folhas ligadas por ramos (roteadores).

A topologia em malha é caracterizada por um rede mais complexas pois permite o roteamento de mensagens entre os dispositivos através de diversas rotas. Esta é sem dúvida uma topologia bastante desejável pois permite grande fiabilidade na comunicação de dispositivos. Assim, em caso de mal funcionamento de um roteador é sempre possível definir outra rota de comunicação entre dispositivos. A desvantagem da utilização desta topologia relaciona-se com o consumo elevado de energia dos dispositivos, isto porque os roteadores têm que estar permanentemente ligados.

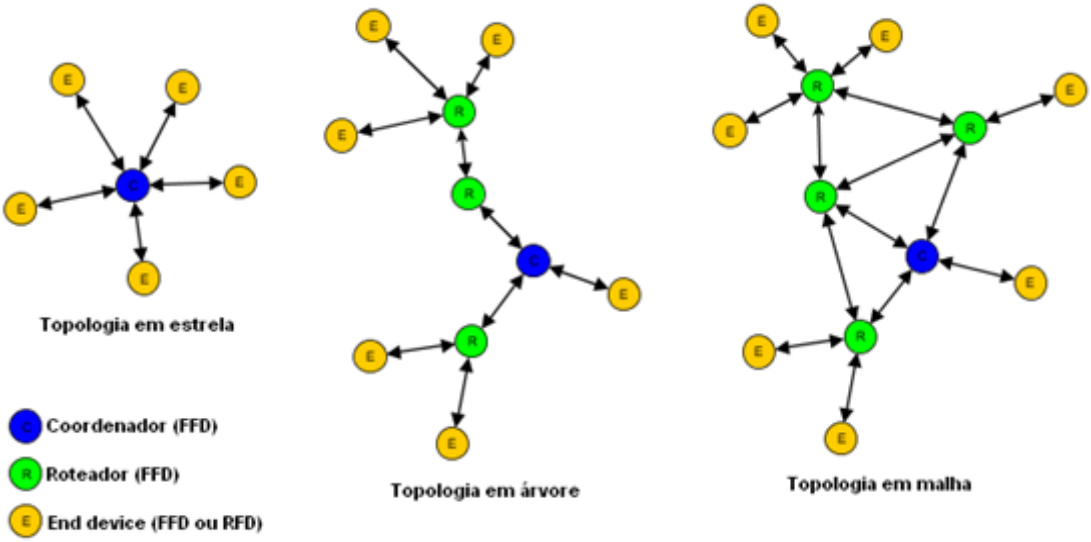

Imagem 4.9: Topologias ZigBee

- **Espaço de endereçamento**

O padrão *IEEE 802.15.4* usa um endereço único de 64 bits para cada dispositivo de rádio. Este endereço é usado nas comunicações ponto a ponto na associação à rede. A partir do momento que a associação é estabelecida, o identificador de 64 bits é substituído por um identificador de 16 bits, reduzindo assim o cabeçalho das tramas transmitidas por e para este dispositivo.

. Atribuição de endereços

Durante o processo de associação de um dispositivo à rede *ZigBee*, o coordenador ou o roteador mais próximo atribui-lhe um endereço de 16 bits segundo um algoritmo estruturado em árvore.

Esta atribuição é feita por uma entidade designada *stack profile*. Esta *stack profile* inclui a definição da profundidade máxima da rede *(maximum network depth)*, número máximo de filhos roteadores *(maximum number of child routers)* e número máximo de filhos *(maximum number of children)*.

4.6.2. Estrutura de uma trama genérico NPDU

O formato da *Network Protocol Data Unit (NPDU)* é constituída por dois campos, o *NWK Header, e NWK Payout.*

O *NPDU* consiste em:

- ***Frame control*** (controlo da trama)- especifica o tipo da trama;
- ***Destination address*** (endereço destino)– endereço destino de 16-bits;
- ***Source address*** (endereço fonte)– endereço fonte de 16-bits;
- ***Radius*** (raio)- alcance da transmissão, em saltos (1 octeto);
- ***Sequence Number Field*** -número identificador da trama transmitida (1 octeto);
- ***Destination IEEE Address Field*** – contém o endereço *IEEE* destino de 64-bits (opcional);
- ***Source IEEE Address Field*** – contém o endereço *IEEE* fonte de 64-bits (opcional);
- ***Multicast Control Field*** – activa a opção de envio *multicast* (1 octeto, opcional);
- ***Source Route Subframe Field*** – activa a opção de rota específica do envio da trama (opcional e tamanho variável);
- ***Frame payload*** - contém a informação de cada trama *NWK* (tamanho variável);

Octets: 2	**2**	**2**	**1**	**1**	**0/8**	**0/8**	**0/1**	**Variable**	**Variable**
Frame control	Destination address	Source address	Radius	Sequence number	Destination IEEE Address	Source IEEE Address	Multicast control	Source route subframe	Frame payload
NWK Header									Payload

Imagem 4.10: Formato de uma trama NWK genérica [5]

4.7 . Camada de Aplicação

A camada de aplicação *ZigBee* contém a sub-camada de suporte à aplicação (APS - *Application Support SubLayer),* objectos de aplicação (*ZDO - ZigBee Device Object) e a Application Framework (AF).*

Esta camada tem como função garantir uma correcta gestão e um suporte fiável para as diversas aplicações [5].

4.7.1. Application Support SubLayer (*APS*)

A *APS* fornece um interface entre a camada *NWK* e a camada de aplicação através de um conjunto geral de serviços que são usados pelo *ZDO* e pelas aplicações definidas pelo cliente/fabricante. Os serviços desta camada são fornecidos pelas seguintes entidades:

APS data entity (APSDE), através do *Application Service Data Entity Access Point (APSDE-SAP).* Esta entidade possibilita a transmissão de dados para o transporte de *PDU's* da aplicação entre dois ou mais dispositivos localizados na mesma rede e uma filtragem das mensagens endereçadas ao grupo. A *APSDE* suporta ainda a fragmentação e reconstrução das mensagens de tamanho superior que o *payload* suportado pelas *Application Service Data Units.*

APS management entity (APSME), através do *Application Service Management Entity Access Point (APSME-SAP).* Esta entidade fornece serviços de segurança, registo e remoção de endereços de grupo e ainda mantém uma base de dados dos dispositivos geridos que tem o nome *APS information base (AIB).*

4.7.2. Application Framework

A *Application Framework* é um ambiente em que os objectos de aplicação (*ZDO*) estão guardados em dispositivos *ZigBee*. Dentro desta *framework* os objectos de aplicação enviam e recebem dados através do *APSDE-SAP*, realizando funções de controlo e manutenção das camadas protocolares do dispositivo *ZigBee* e inicialização de funções de rede. O serviço de dados utilizado por estes objectos inclui funções de pedido, confirmação, resposta e primitivas de indicação para a transferência de dados, que são utilizados para indicar a transferência de dados da *APS* para a aplicação ou entidade de destino. As funções de pedido suportam transferências de dados entre aplicações de

entidades de objectos.

4.7.3. ZigBee Device Objects

Os *ZigBee Device Objects* representam uma base de funcionalidades que fornece um interface entre os objectos de aplicação, o perfil do dispositivo e a *Application Support Sublayer*. Estes objectos têm o objectivo de satisfazer os requisitos de todas as aplicações que estejam a ser executadas na pilha protocolar *ZigBee*. Os *ZDO* são responsáveis por inicializar a *APS*, a camada *NWK* e o serviço de segurança, permitindo definir a tarefa do dispositivo na rede (coordenador, roteador ou *end device*). Estes objectos têm interfaces públicas com os objectos da *Application Framework* para que estes possam fazer o controlo de funções de dispositivo da rede.

4.7.4. Estrutura de uma trama genérica *APDU*

A *Application Protocol Data Unit (APDU)* é constituída por dois campos, o *APS Header, e APS Payout* [5].

O campo *APS Header* consiste em:

- ***Frame control*** (controlo da trama)- especifica o tipo da trama;
- ***Destination enpoint*** – identifica a aplicação do dispositivo destino (*8 bits*);
- ***Group address*** – identifica se a trama será enviada para um grupo de *endpoints* assim como esses grupos *(16 bits)*.
- ***Cluster identification*** –identificador da trama transmitida (*16 bits*);
- ***Source enpoint*** -identifica a aplicação de operação do dispositivo fonte (*8bits*);
- ***APS counter*** – identificador da mensagem, previne a recepção duplicada de mensagens (*8bits*);
- ***APS payload*** -contém a informação de cada trama *APS* (tamanho variável);

<table>
<tr><td>Octets:
1</td><td>0/1</td><td>0/2</td><td>0/2</td><td>0/2</td><td>0/1</td><td>1</td><td>Variable</td></tr>
<tr><td rowspan="2">Frame control</td><td>Destination endpoint</td><td>Group address</td><td>Cluster identifier</td><td>Profile Identifier</td><td>Source endpoint</td><td rowspan="2">APS counter</td><td rowspan="2">Frame payload</td></tr>
<tr><td colspan="5">Addressing fields</td></tr>
<tr><td colspan="7">APS header</td><td>APS payload</td></tr>
</table>

Imagem 4.11: Formato de uma trama APS genérica [5]

Capítulo 5

Antenas impressas

Neste capítulo são apresentados os fundamentos teóricos das antenas impressas *(microstrip antennas) [33].* Na secção 2 são referidas as principais características das antenas e na secção 3 são apresentadas as vantagens e desvantagens de antenas impressas. Na secção 4 são referidos os passos para projectar uma antena *microstrip* rectangular e na secção 5 são apresentadas as simulações efectuadas para as antenas projectadas. Na secção 6 são referidos os resultados medidos e uma comparação com os resultados teóricos. A secção 7 encerra o capítulo descrevendo as conclusões das simulações efectuadas.

5.1 . Estrutura e características de uma antena impressa

Uma antena impressa é construída através de técnicas de fabrico de circuitos impressos. De um modo simplificado uma antena impressa consiste em duas placas condutoras paralelas separadas por um dieléctrico, designado substrato. O substrato é

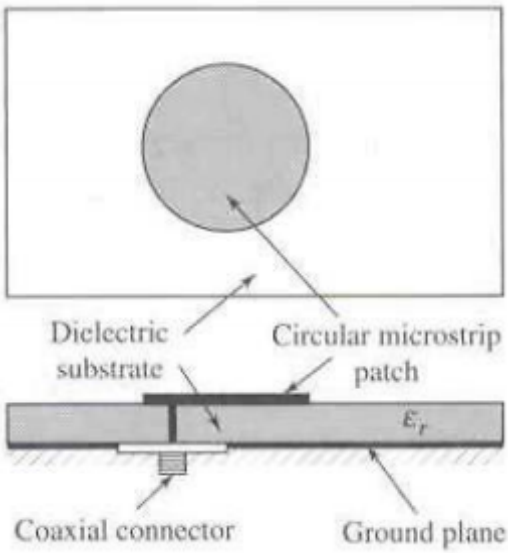

Imagem 5.1: Exemplo de antena microstrip [33]

caracterizado por uma altura *h*, uma constante dieléctrica (ε_r) e uma tangente de perdas (*tan δ*). A placa superior é o elemento radiante *(patch)* e a inferior designa-se por plano massa.

O comprimento típica da *patch (L)* é compreendido entre $\lambda/3 < L < \lambda/2$, onde λ representa o comprimento de onda do sinal que se propaga na estrutura planar. Os substratos usualmente têm constante dieléctrica entre 2.2 e 12 e a tangente de perdas apresenta valores entre 0,0002 e 0.02 [33][47].

Existem várias geometrias que a *patch* pode assumir, de acordo com a aplicação pretendida. Devido à simplificação de análise, fabrico e características atractivas de radiação, as mais utilizadas são as de geometria quadrada, rectangular e circular.

Existem vários modos de alimentar uma antena *microstrip*, através de: linha *microstrip*, cabo coaxial, linha de fenda e acoplamento electromagnético [33].

5.2 . Parâmetros fundamentais das antenas

As antenas representam para um sistema sem fios a interface com o meio, sendo estas o primeiro componente do sistema em caso de recepção ou último em caso de transmissão do sinal. Os principais parâmetros que caracterizam uma antena são: diagrama de radiação, directividade, ganho, impedância de entrada e largura de banda.

5.2.1. Diagrama de radiação

O diagrama de radiação pode ser definido como a representação gráfica das propriedades radiantes das antenas, em função da direcção. Estas propriedades dizem respeito à amplitude, fase e polarização do campo electromagnético. O diagrama de radiação refere-se a uma representação tridimensional, mas habitualmente é representado por vários cortes bidimensionais. Os cortes usualmente utilizados referem-se ao plano *E* (plano formado pela direcção do vector campo eléctrico e direcção de máxima radiação) e plano *H* (plano formado pela direcção do vector campo magnético e direcção de máxima radiação).

5.2.2. Ganho e directividade

Estes dois parâmetros são essenciais para o projecto de uma antena pois descrevem a distribuição da radiação da antena pelo espaço em torno desta. Uma antena que radia de igual modo para todos as direcções, terá directividade igual a 1 e designa-se por antena isotrópica. Uma antena de alta directividade é aquela que concentra num pequeno ângulo sólido toda a potência radiada.

A directividade representa um índice numérico que mede a capacidade de um antena concentrar a potência radiada na direcção de máxima radiação ou concentrar a absorção de potência incidente numa determinada direcção no caso de antenas receptoras.

Se considerarmos P_0 a potência fornecida por um gerador ligado aos terminais de uma antena e e_{ant} a eficiência desta, a potência radiada será:

$$P_r = e_{ant} P_0$$

Caso a potência seja radiada isotropicamente (de igual modo em todas as direcções) a intensidade de radiação $U(\theta, \phi)$ seria $P_r / 4\pi$ [33].

Na prática a potência não é radiada isotropicamente, deste modo, a intensidade da radiação numa dada direcção (θ, ϕ) é definida como:

$$U(\theta, \varphi) = \frac{P_r}{4\pi} D(\theta, \varphi)$$

onde $D(\theta, \phi)$ representa a directividade na direcção (θ, ϕ) .

Esta fórmula pode também ser reescrita na forma seguinte:

$$D(\theta,\varphi) = 4\pi \frac{U(\theta,\varphi)}{P_r}$$

Verifica-se então que a directividade é a razão entre a intensidade da radiação da antena numa determinada direcção e a intensidade de radiação de uma antena isotrópica radiando a mesma potência.

O ganho é definido de forma semelhante, como a razão entre a potência radiada numa determinada direcção (θ,ϕ) e a potência que seria radiada por uma antena isotrópica sem perdas com a mesma potência de entrada P_0 :

$$G(\theta,\varphi) = 4\pi \frac{U(\theta,\varphi)}{P_0}$$

Relacionando as equações anteriores pode-se escrever:

$$\frac{G(\theta,\varphi)}{D(\theta,\varphi)} = \frac{P_r}{P_0} = e_{ant}$$

Ganho e directividade estão relacionadas entre si através do factor de eficiência de uma antena, sendo este parâmetro nunca superior à unidade. Deste modo, o ganho é sempre inferior ou igual à directividade.

5.2.3 Impedância de entrada

A impedância que uma antena apresenta à linha de transmissão que a alimenta designa-se por impedância de entrada (*Zin*) . Se a antena for considerada um elemento de um sistema, esta pode ser vista como uma carga numa linha de alimentação, e como tal representada pela sua impedância de entrada. Para tal é necessário determinar a corrente que a atravessa e a tensão aos seus terminais, sendo a impedância de entrada calculada através do quociente entre a tensão e a corrente *(Lei de Ohm*). A eficiência (e_{ant}) da antena é calculada como a razão entre a resistência de radiação (R_r) e a soma desta com a resistência de perdas (R_p) . Geralmente descritas as equações como:

$$Z_{in} = R + jX = \frac{V(f)}{I(f)},\ com\ R = R_r + R_p\ \ e\ e_{ant} = \frac{R_r}{R_r + R_p}$$

5.2.4 VSWR

Para qualquer ponto de uma linha de transmissão a tensão e a corrente resultam de duas ondas (incidente e reflectida) que se propagam em sentidos opostos. O coeficiente de reflexão, ρ, pode ser definido como a razão entre a amplitude da onda reflectida e a amplitude da onda incidente de tensão num determinado ponto. Designando a impedância de carga por Z_L e a impedância característica da linha por Z_0 , então o coeficiente de reflexão pode ser definido como:

$$\rho = \frac{Z_L - Z_0}{Z_L + Z_0}$$

Se a linha estiver terminada com uma impedância de carga $Z = Z_L = Z_0$ só existiria uma onda, a onda incidente, o que implicaria um coeficiente de reflexão nulo. Caso se verificasse uma desadaptação do sistema $(Z_L \neq Z_0)$ existiria além da onda incidente uma onda reflectida, sendo a interferência entre as duas ondas conhecida como onda estacionária.

Por definição, o coeficiente de onda estacionária, *VSWR (Voltage Standing Wave Ratio)* é a razão entre os valores máximo e mínimo da envolvente num sistema com onda estacionária e é definido como:

$$VSWR = \frac{V_{\max}}{V_{\min}} = \frac{1 + |\rho|}{1 - |\rho|}$$

Num sistema sem onda reflectida o *VSWR* é igual à unidade. Quanto maior a desadaptação do sistema, maior o valor de *VSWR*. Normalmente são de interesse as frequências onde o valor de *VSWR* é inferior ou igual a 2.

5.2.5 Largura de Banda

A largura de banda de uma antena diz respeito à faixa de frequência no qual a antena opera, em torno da sua frequência de operação central (frequência de ressonância) e que satisfaz determinados parâmetros de funcionamento. Pode-se designar então a largura de banda como a gama de frequências onde as características da antena (tais como impedância de entrada, diagrama de radiação ou *VSWR*) se encontram dentro de uma gama aceitável quando comparados com as características à frequência central [33].

A largura de banda pode ser apresentada de modo percentual ou em relação ao

posicionamento de frequências (inferior e superior).

O primeiro caso é adequado quando a largura de banda é muito menor do que a frequência de operação central. Sendo calculada através da seguinte fórmula:

$$LB = \frac{f_H - f_L}{f_C} \times 100$$

onde f_H representa a frequência superior da banda, f_L a frequência inferior e f_C a frequência central da banda. Um dos critérios mais usado para a determinação da largura de banda poderá ser a gama de *VSWR* inferior a 2 ou um valor de perdas por retorno *(S11-return loss)* inferior a *-10dB*.

O segundo caso é utilizado quando a frequência superior se apresentar maior ou igual ao dobro da frequência inferior da banda. Por exemplo, uma largura de banda 5:1 indica que a frequência superior é cinco vezes maior que a frequência inferior da banda.

5.3 . Vantagens e limitações

As antenas *microstrip*, comparativamente a antenas de microondas tradicionais apresentam várias vantagens, entre elas[33][22]:

- tamanho e peso reduzido;
- fabrico simplificado e custo reduzido;
- diversidade de formatos;
- diagramas de radiação de várias formas;
- fácil utilização para *array* de antenas.

As desvantagens deste tipo de antenas estão relacionadas com:

- largura de banda reduzida;
- ganho reduzido;
- grande sensibilidade ao substrato utilizado, onde permitividade dieléctrica elevada implica baixa eficiência e largura de banda reduzida;
- excitação de ondas de superficiais que diminuem a eficiência da antena.

Conforme a aplicação desejada poder-se-á utilizar métodos de modo a melhorar certos parâmetro em troca do menor desempenho de outros. Por exemplo, caso se deseje uma maior largura de banda, poder-se-á aumentar a altura do substrato. Mas em troca

desta melhoria, serão introduzidas ondas superficiais indesejáveis, que irão alterar as características do diagrama de radiação e da polarização da antena [33].

5.4 . Dimensionamento de uma antena *microstrip* rectangular

Nesta dissertação foi dimensionada uma antena *microstrip* com objectivo de melhorar a exactidão do sistema de localização comparativamente ao resultado obtido através do uso dos dipolos de meia onda (*Titanis 2,4GHz*) [49] fornecido no *kit* de desenvolvimento.

Para dimensionar um antena *microstrip* deve-se especificar em primeiro lugar a frequência de ressonância (f_r) , a altura do substrato (*h*), e a permitividade eléctrica relativa deste (ε_r) [33].

No nosso caso foram especificados os seguintes parâmetros:

- ***f_r = 2,44 GHz***
- ***ε_r= 4,7 (típico do FR4)***
- ***h = 3,2 mm***

A frequência de ressonância considerada foi de 2,44*GHz* visto ser a frequência central de funcionamento da tecnologia *ZigBee,* deste modo é possível a sua utilização para qualquer uma dos 16 canais da banda 2,4*GHz*. Foi utilizado o último canal (canal 26) para funcionamento do sistema a uma frequência de 2.48*GHz*.

Os passos para calcular as dimensões da antena *microstrip* segundo [48] são os seguintes:

Calcular a largura da patch (W)

$$W = \frac{v_0}{2 f_r}\sqrt{\frac{2}{\varepsilon_r + 1}}$$

vo - velocidade da luz no vazio $3\text{x}10^8 (m/s)$.

Calcular a permitividade eficaz (ε_{reff})

$$\varepsilon_{reff} = \frac{\varepsilon_r + 1}{2} + \frac{\varepsilon_r - 1}{2}\left[1 + \frac{12h}{W}\right]^{-\frac{1}{2}}$$

Calcular incremento da largura da patch (Δ_L)

$$\Delta_L = 0.412h\frac{\left(\varepsilon_{reff} + 0.3\right)\left(\frac{W}{h} + 0.264\right)}{\left(\varepsilon_{reff} - 0.258\right)\left(\frac{W}{h} + 0.8\right)}$$

Calcular comprimento da patch (L)

$$L = \frac{v_0}{2f_r\sqrt{\varepsilon_{reff}}} - 2\Delta_L$$

Calcular comprimento do plano massa (L_g)

$$L_g = 6h + L$$

Calcular largura do plano massa (W_g)

$$W_g = 6h + W$$

Dimensões obtidas

Através das fórmulas descritas anteriormente foi possível calcular as dimensões da *patch* e plano massa representadas pela Imagem 5.2.

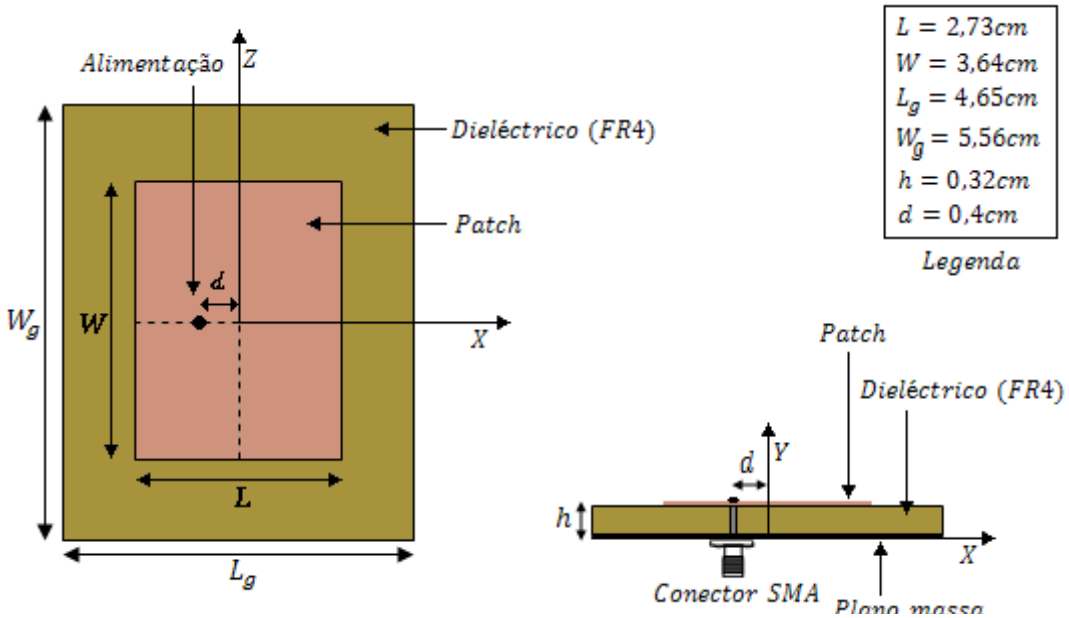

Imagem 5.2: Dimensionamento da antena projectada

A posição da alimentação foi estimada através do simulador *HFSS 9.2 da Ansoft*, onde a posição que apresentou menores perdas por retorno (*S11 – return loss*) representa a posição para melhor adaptação da antena.

Nas simulações efectuadas a posição com menores perdas por retorno foi a 4mm para a esquerda do centro de L, como mostrado na Tabela 5.1.

Posição em relação ao centro de L (cm)	***Return loss S11 (dB)***
-0.35	-19
-0.4	-31
-0.45	-21
-0.5	-16

Tabela 5.1: Posição da alimentação da patch

5.5 . Simulações realizadas

Através do simulador *HFSS* foram efectuadas várias simulações de modo a estimar a gama de variação da permitividade relativa que garantisse um correcto funcionamento da antena *microstrip* dimensionada e representada na Imagem 5.3.

O conhecimento desta gama é importante visto a permitividade do dieléctrico poder apresentar variações ao longo do comprimento deste.

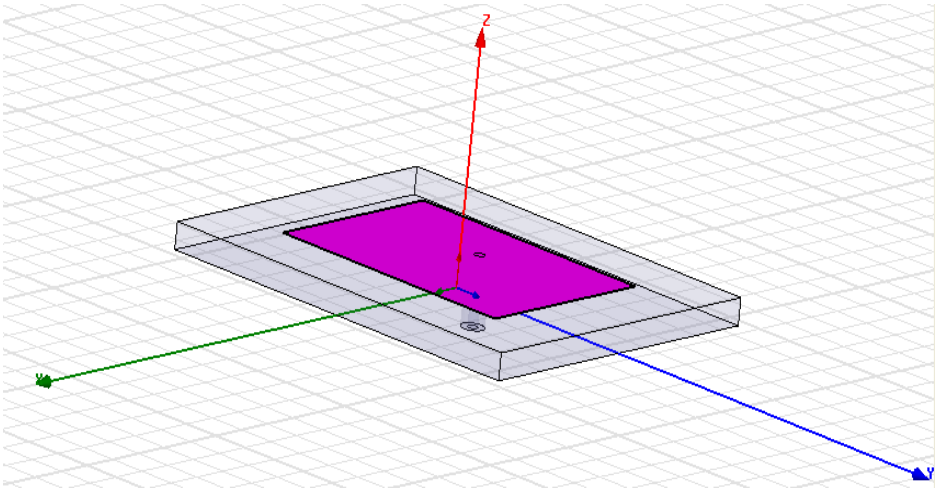

Imagem 5.3: Representação da antena microstrip no simulador

A largura de banda verificada foi cerca de 100*MHz* (4,1%), considerando o limite S11 = -10dB e frequência central de 2,44GHz.

O limite inferior de ε_r permitido para um bom funcionamento do sistema para o

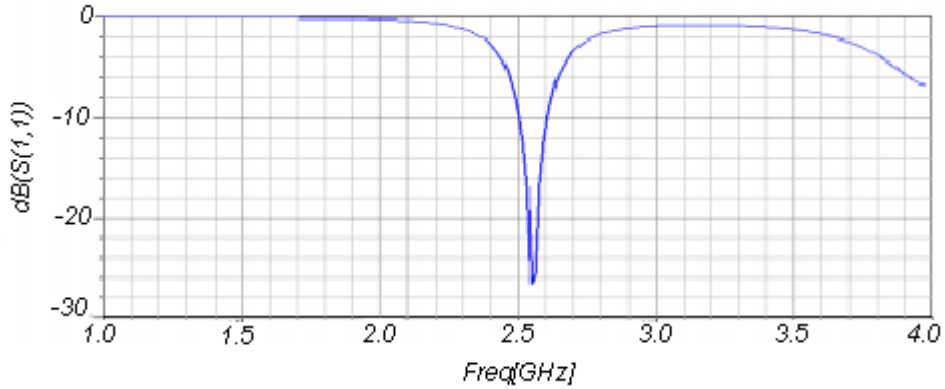

Imagem 5.4: S11 para εr = 4.4

canal (26) a *2.48GHz* é de 4.4 como representado na Imagem 5.4.

O limite máximo de ε_r permitido para um bom funcionamento do sistema para ~*2.48GHz* é de 4.9 como representado na Imagem 5.5.

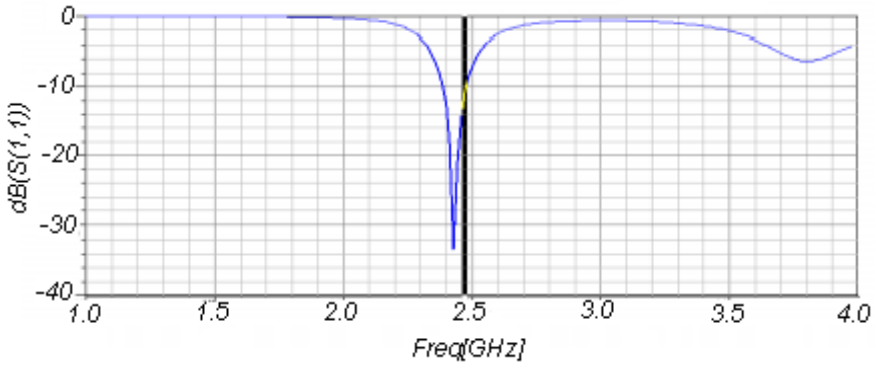

Imagem 5.5: S11 para εr = 4.9

Diagrama de radiação

Por fim foram simulados os diagramas de radiação da antena projectada. As antenas foram simuladas com ε_r = 4.7, frequência *2.48GHz* e posição da alimentação a -0.4 cm do centro de L. Nas imagens 5.6 e 5.7 são representadas os diagramas de radiação das antenas simuladas e projectadas em dois planos ortogonais ($\varphi=0^o$ e $\varphi=90^o$). A análise das antenas projectadas foi efectuada numa câmara anecóica. É de referir que a rotação da antena *microstrip* no processo de análise não foi total (de 360^o), esta é a razão de não haver registo de valores entre 180^o e 195^o.

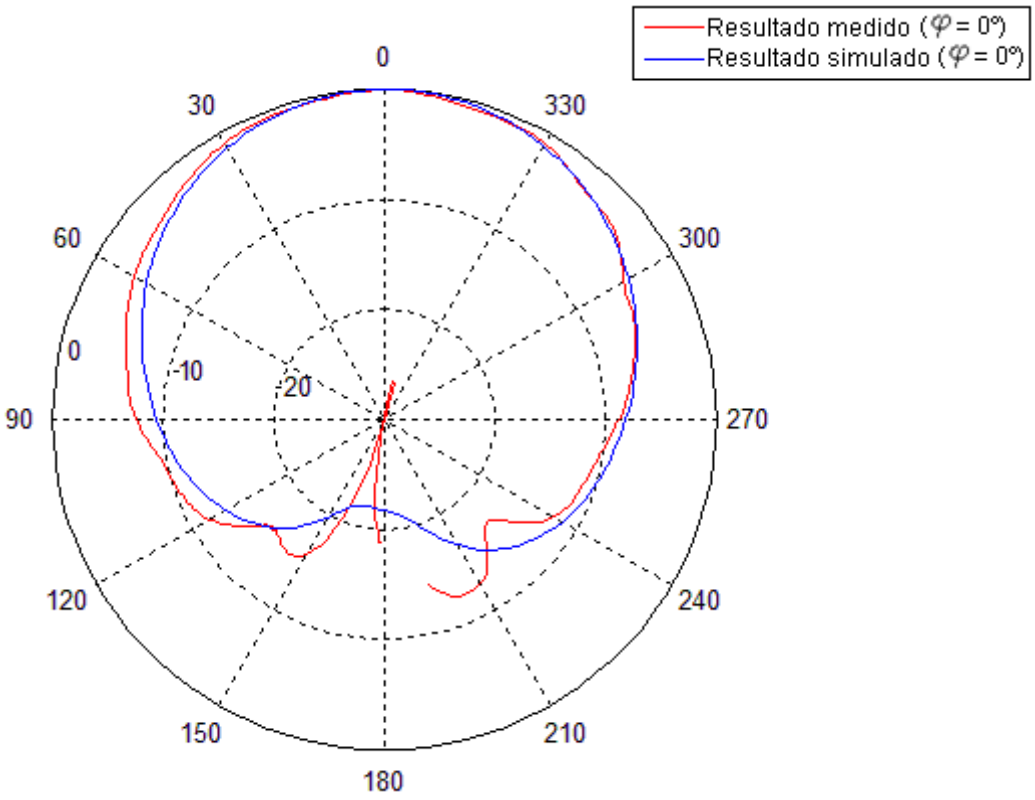

Imagem 5.6: Comparação do diagrama de radiação medido/simulado para φ=0º

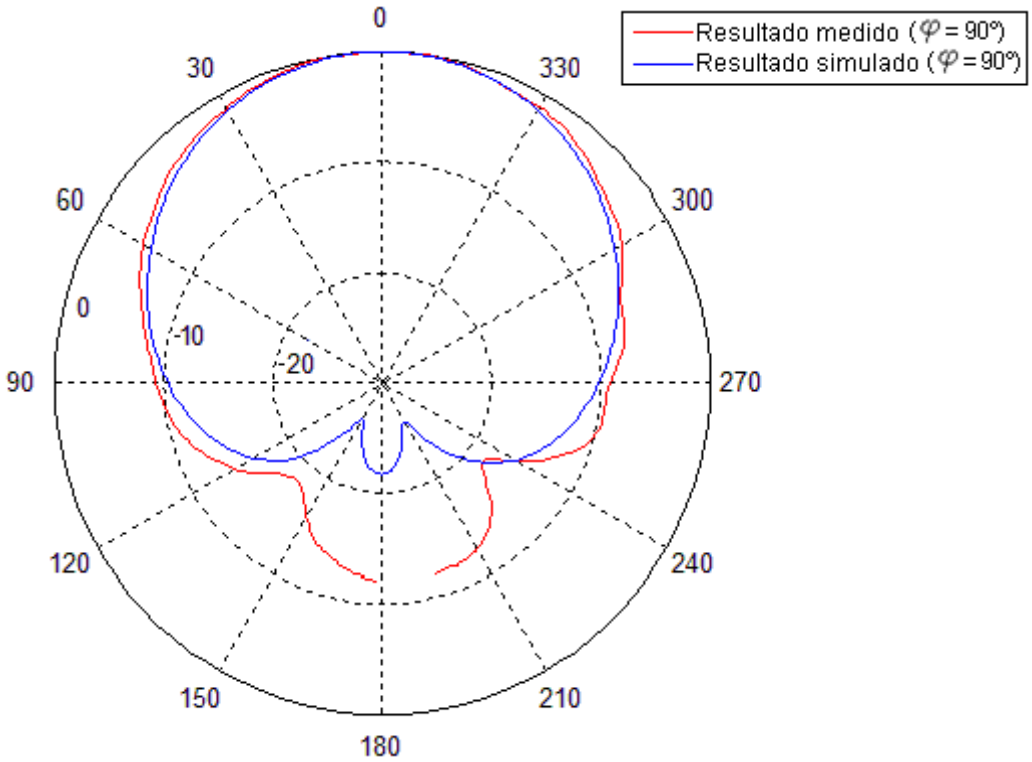

Imagem 5.7: Comparação do diagrama de radiação medido/simulado para φ=90º

5.6 . Medições das antenas projectadas

As antenas *microstrip* projectadas, representadas pela Imagem 5.8 e de dimensionamento representado na Imagem 5.2 registaram valores muito aproximados dos esperados e um funcionamento muito eficaz para o sistema implementado.

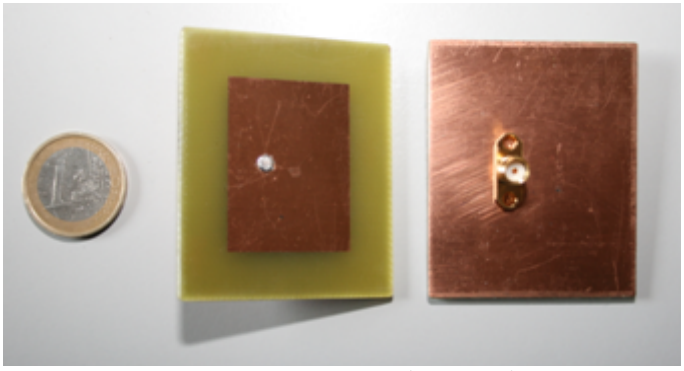

Imagem 5.8: Antenas Patch projectadas

O valor de *S11* foi registado através de um analisador de redes vectorial e apresentou um valor aproximado de -23*dBs* para uma frequência central de 2,444*GHz,* muito próximo dos 2,44*GHz* esperados. Esta análise é representada pela Imagem 5.9. A largura de banda registada foi cerca de 3,3% *(~80MHz),*valor aproximado do esperado, *4,1%*. Esta largura de banda possibilita o funcionamento correcto para todos os canais *ZigBee* da banda *ISM* 2,4*GHz.*

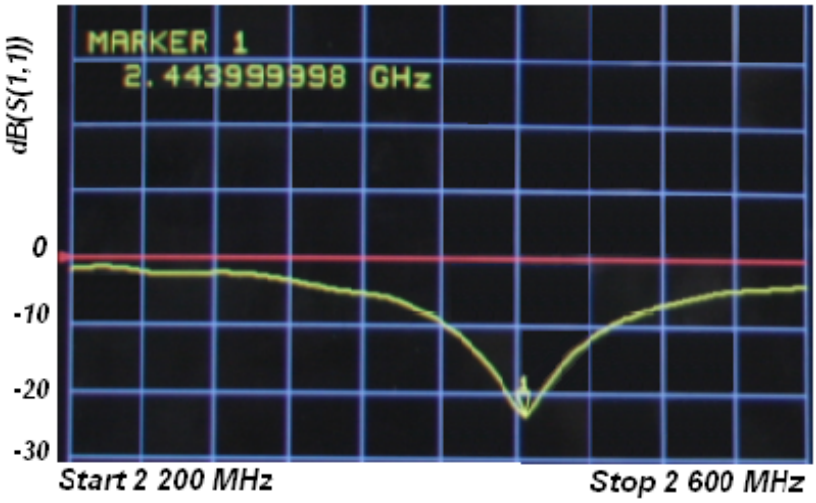

Imagem 5.9: S11 de uma antena microstrip num analisador de redes vectorial

O ganho da antena *microstrip* foi calculado através da comparação do seu valor de ganho com o valor de ganho de uma antena calibrada. É de referir que os valores de calibração da antena apresentavam um limite inferior de 2500*MHz* com um valor de 8,30*dBi*. Foi assumido um valor de 8*dBi* para 2440*MHz*. Foi medido e registada a diferença de ganho entre a antena calibrada e a antena projectada para o máximo de potência através de num voltímetro vectorial. Essa diferença foi então subtraída ao ganho de 8*dBi* da antena calibrada sendo assim possível estimar o valor de ganho da antena projectada. O valor obtido foi de *4,03dBi* muito próximo do valor de *4,42dBi* registados com o simulador.

5.7 . Conclusões

As antenas projectadas apresentaram valores muito próximos dos esperados, entre eles: uma frequência central de 2.444*GHz* muito próximo dos 2,44*GHz* obtidos na simulação; uma largura de banda de 3,3%, suficiente para o funcionamento de qualquer canal *ZigBee* da banda 2,4*GHz;* uma gama aceitável para variação de ε_r (entre 4.4 e 4.9) do dieléctrico para uma frequência de 2,48*GHz (*frequência do canal utilizado*);* e um ganho de 4,03*dBi*..

Estas antenas possibilitaram uma melhoria no desempenho do sistema de localização por proximidade em cerca de 25-30%, como descrito com maior detalhe em 9.2 .

Capítulo 6

Plataformas de Hardware e Software

Neste capítulo são apresentadas as plataformas que serviram de suporte para o desenvolvimento desta dissertação. Inicialmente será apresentado o *kit* de desenvolvimento utilizado assim como uma descrição detalhada dos seus componentes. Posteriormente serão referidas as antenas desenvolvidas no âmbito desta dissertação e por fim serão descritos os programas utilizados assim como as suas funções.

6.1 . Hardware

Para sistemas de localização sem fios o tamanho, custo e possibilidade de programação do *firmware* dos dispositivos são requisitos essenciais. A plataforma de desenvolvimento utilizada neste projecto foi a *CC2431ZDK da Texas Instruments*, pois apresentava todas essas características.

O *hardware* disponibilizado por esta plataforma é representada pela Imagem 6.1 e descrita de um breve modo na Tabela 6.1.

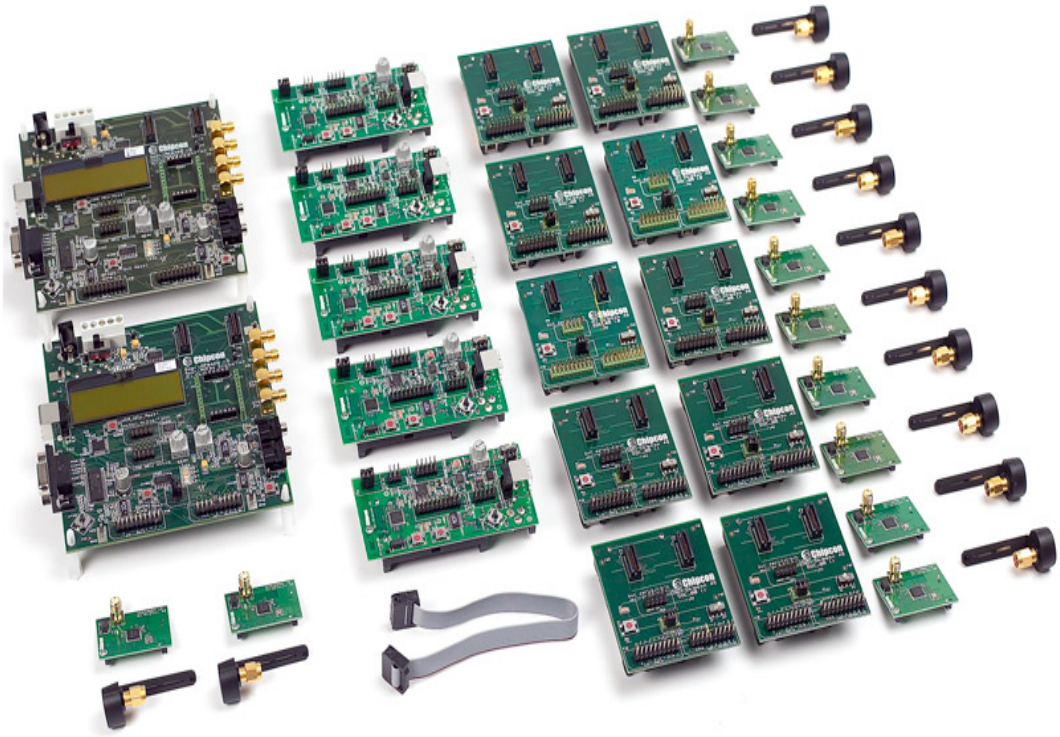

Imagem 6.1: Plataforma de desenvolvimento CC2431ZDK[46]

Quantidade	Módulo	Descrição
2	*SmartRF04EB Evaluation Board*	*Placa de desenvolvimento e configuração dos módulos ZDK CC2430/31*
2	*CC2430EM Evaluation Modules*	*Módulos SoC ZDK CC2430*
5	*CC2431DB Development Boards*	*Não utilizado*
10	*Battery Boards para uso com CC2431EMs*	*Suporte para módulos CC2430/31*
10	*CC2431EM Evaluation Modules*	*Módulos de comunicação*
12	*Titanis 2.4GHz SMA*	*Antenas dipolo de meio comprimento de onda*
2	*Cabo USB*	*Cabo de programação dos SoCs e alimentação da placa SmartRF04EB*
1	*Cabo RS-232*	*Suporta as comunicações RS-232 entre o PC server e o gateway*

Tabela 6.1: Breve descrição dos componentes da plataforma ZDK2431[27]

6.1.1. Módulo *CC2430/31*

Imagem 6.2: Módulos CC2430/31[27]

O *SoC (System-on-Chip) CC2430/31* é um módulo especialmente criada para redes sem fios de baixo consumo baseadas em *IEEE 802.15.4 e ZigBee*.

Este módulo contém um transceptor *CC2420*, um micro-controlador de baixo consumo *8051, 8kB de RAM, 128kB* de memória *flash* e muitas outras características descritas na Tabela 6.2.

O *MCU 8051* é um micro controlador de alta performance e baixo consumo que corre a *Z-Stack* a partir da memória *flash*. Através desta *Z-Stack*, o *CC2430/31* pode ser configurado para utilizar a *stack ZigBee* para comunicar com os vários elementos da rede.

As características principais destes módulos são descritas na Tabela 6.2.

Memória Flash programável de 128kB;
4 Modos de consumo para reduzir o consumo energético
Memória SRAM de 4kB com retenção de dados em todos os modos de funcionamento
Memória adicional SRAM de 4kB com retenção de dados para o modo 0 e 1
Transição rápida entre os diferentes modos
8 canais ADC com resolução configurável
Coprocessador de segurança AES
Suporte de CSMA/CA via hardware
4 timers:1 IEEE 802.15.4 MAC timer, um timer de 16bit e 2 de 8bits
Duas USART's programáveis para operação master/slave SPI ou UART
21 pinos I/O configuráveis
Suporte digital de RSSI / LQI
Suporta 16 canais rádio programáveis entre 2,4GHz e 2,4835GHz
Sensibilidade excelente -94dBm
Elevada rejeição de canal adjacente
Taxa de transmissão de 250Kbps
Consumo de cerca de 31mA em RX, 29mA em TX e 0.5µA em Modo Sleep
Alimentação de 2.0 a 3.6 V (power Modes)

Tabela 6.2: Características do módulo CC2431/30[15]

6.1.2. Módulo CC2430/31SoC BB (Battery Board)

A Imagem 6.3 mostra o *System on Chip battery board (SoC_BB)*. As funções principais desta placa são alimentar o módulo *CC2430EM ou CC2431EM* através de duas pilhas *AA* e o suporte para uma análise dos sinais do módulo inserido.

Os conectores *I/O A* e *B* permitem o acesso a todos os *pins I/O do SoC* inserido.

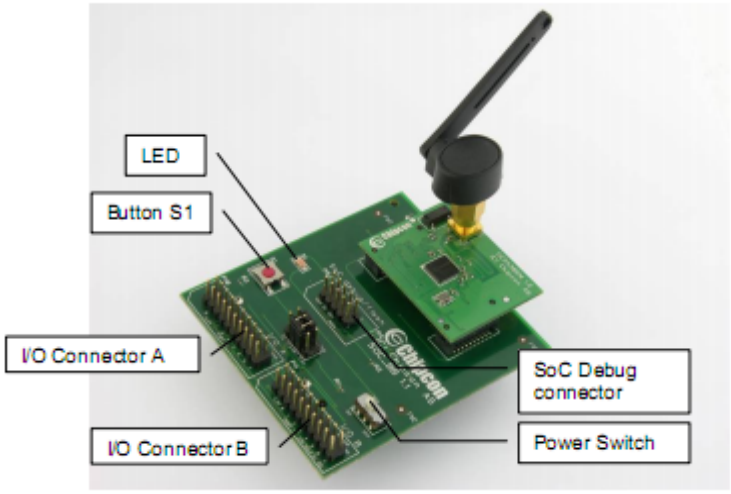

Imagem 6.3: Módulo CC2431EM suportado por uma placa SoC_BB[27]

O conector *SoC Debug* pode ser usado para descarregar o *firmware* para a memória *flash* do módulo inserido e para executar a depuração do mesmo.

O *Power Switch* permite ligar ou desligar o *SoC_BB*.

O *button S1* (programável) está inicialmente programado para reiniciar o módulo inserido no *SoC_BB*.

A *LED* permite uma indicação do estado das baterias, se apresentar uma cor vermelha contínua indica o funcionamento correcto das baterias, em caso de ser intermitente indica bateria fraca.

6.1.3. SmartRF04EB

O *SmartRF04EB* é a plataforma que funciona como *gateway* entre a rede de sensores sem fios e o *PC* a que está conectado por *RS-232* ou *USB*.

Esta plataforma é que permite a programação dos módulos *(CC2430EM e CC2431EM)* através no programa *IAR* assim como o suporte para o funcionamento do programa *Packet Sniffer e IEEE Adress Programmer*.

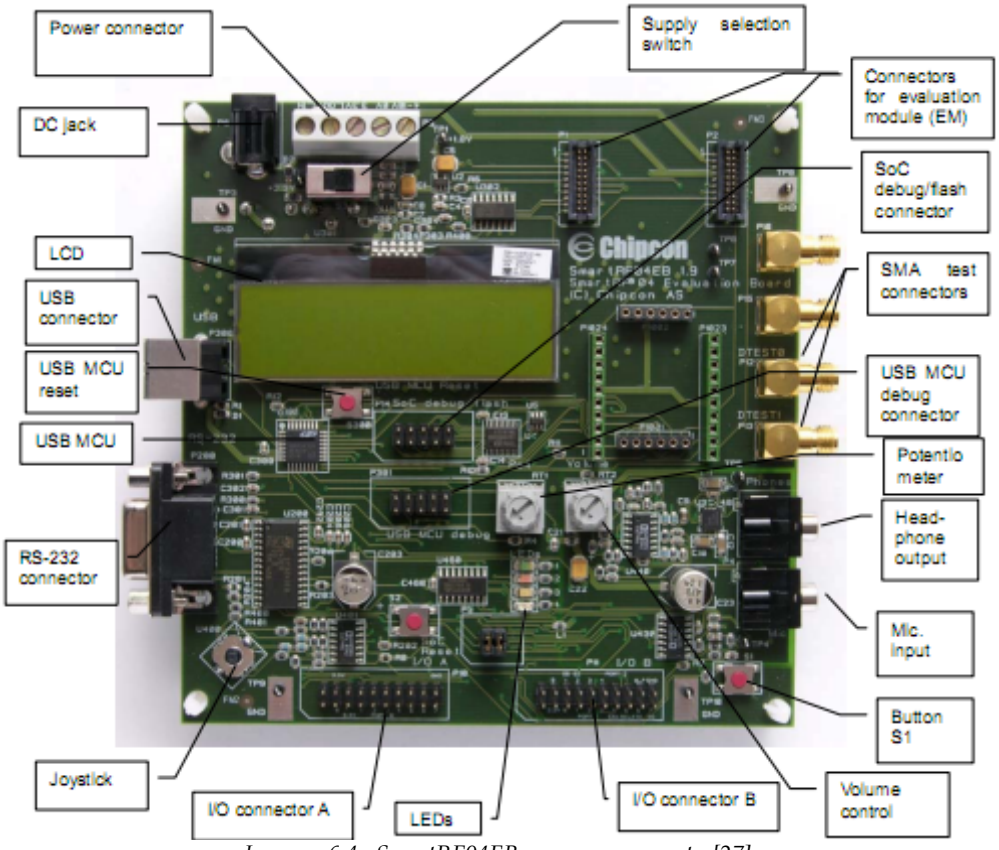

Imagem 6.4: SmartRF04EB e seus componentes[27]

Características	***Descrição***
Alimentação	*Pode ser efectuada através do DC Jack, power connectors e via USB; contém um regulador para 3.3V*
Interface USB	*Permite a interface do SmartRF04EB com o PC possibilitando a programação e debug dos módulos incorporados nesta*
Interface RS-232	*Permite a interface de comunicação com as aplicações executados no PC*
Interface com o utilizador	*É disponibilizado um Joystick e um botão que permitem a execução de tarefas programadas. 4 leds e um LCD possibilitam a indicação de várias informações sobre o estado dos módulos*
Interface Áudio	*Um microfone (input) permite despoletar interrupções, como por exemplo acordar o módulo de Sleep Mode. O Head-phone (output) permite a emissão de um sinal quando configurado*
Conectores I/O	*Permitem o acesso a todos os pins de I/O com os módulos inseridos na placa*
EM connectors	*Permitem a ligação dos módulos EM à placa SmartRF04EB*

Tabela 6.3: Características da placa SmartRF04E[27]

6.2 . Software

Nesta secção será apresentado o *software* utilizado para o desenvolvimento desta dissertação.

6.2.1. IAR embedded Workbench MCS-51 8051

O programa *IAR Embedded Workbench MCS-51 8051* é um ferramenta de desenvolvimento que permite efectuar a compilação e depuração de *firmware* para os módulos *CC2430/31* através da placa *SmartRF04EB*;

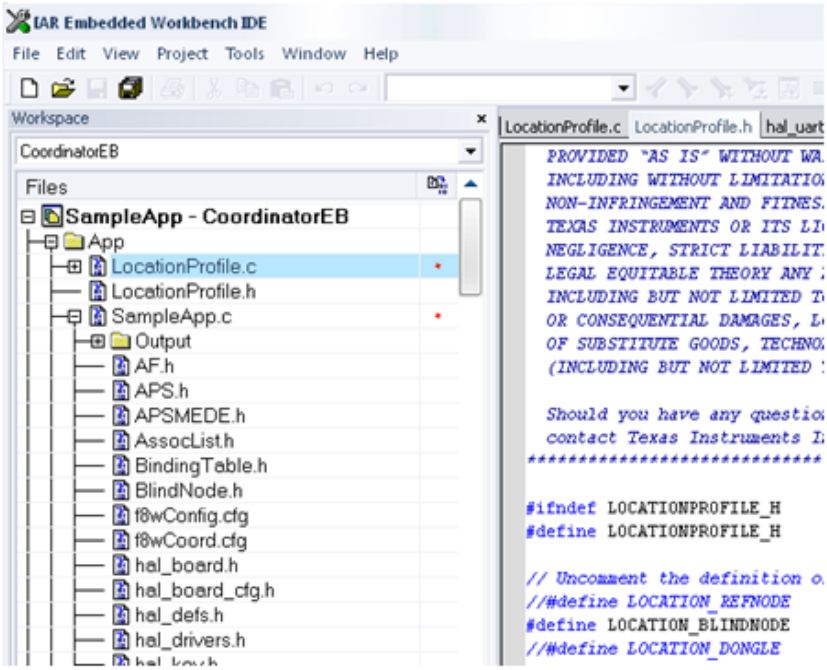

Imagem 6.5: IAR Embedded Workbench MCS-51 8051

6.2.2. Z-Stack v. 1.4.3

Z-Stack v. 1.4.3, é uma ZigBee Compliant Plataform (ZCP) certificada pela *ZigBee Alliance*. A *Z-Stack* inclui uma *Application Programming Interface (API)* e as livrarias: *Hardware abstraction layer (HAL); Operation system abstration layer (OSAL); Stack+IEEE 802.15.4 MAC; User Application e Monitor Test (MT).*

Esta pilha é disponibilizada gratuitamente no site da *TI* e serviu de base para a programação realizada nos módulos do sistema implementado permitindo ao utilizador preocupar-se apenas com a programação da camada de aplicação.

6.2.3. GeneralPacket Sniffer

O *General Packet Sniffer* inclui várias aplicações entre elas o *Sniffer CC2430 IEEE 802.15.4*, que permite monitorizar as mensagem *ZigBee* enviada na rede no alcance de um módulo *CC2430EM*, colocado numa *Evaluation Board.*

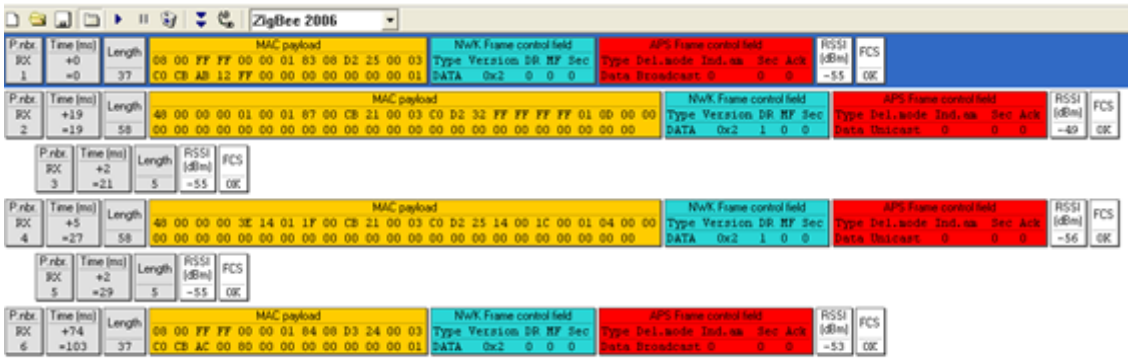

Imagem 6.6: Packet Sniffer 2.7.1

Permite a filtragem de mensagens especificando o canal de escuta assim como a informação que pretendemos visualizar.

6.2.4. IEEE Address Programmer

A actualização do *firmware* dos módulos é um processo que limpa a memória dos módulos. O *MAC address* usualmente é um endereço fixo sem possibilidade de ser alterado.

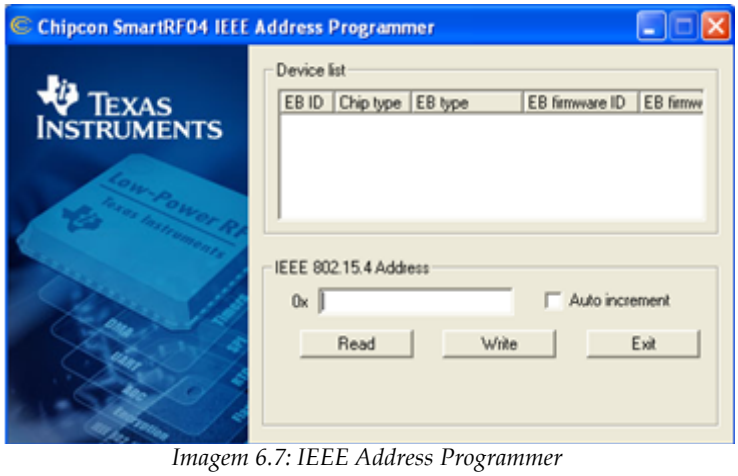

Imagem 6.7: IEEE Address Programmer

O *IEEE Address Programmer* possibilita o registo e alteração do endereço *MAC* nos módulos de acordo com um endereço válido inserido pelo programador.

6.2.5. NetBeans IDE 6.5

O NetBeans IDE é um *ambiente de desenvolvimento integrado (IDE)* gratuito e *open source* que possibilita o suporte para o desenvolvimento de aplicações *Java*. Este foi o programa de suporte para o desenvolvimento da interface gráfica desenvolvida no âmbito desta dissertação.

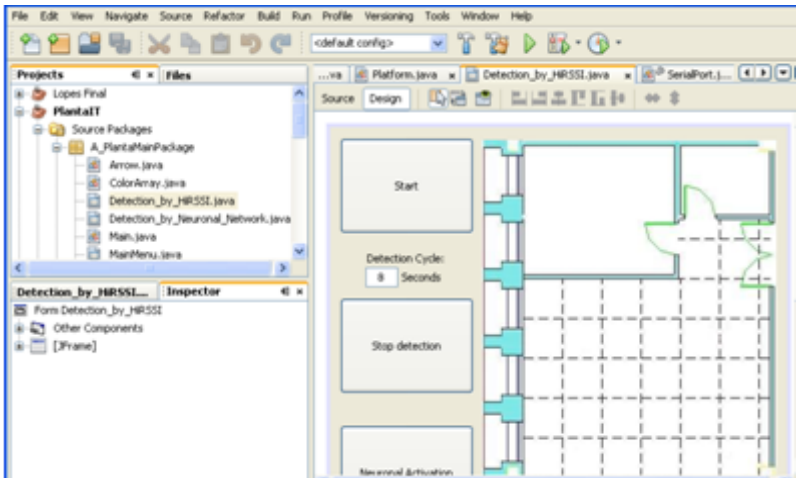

Imagem 6.8: NetBeans IDE

Capítulo 7

Arquitectura do Sistema

Neste capítulo é descrita a arquitectura global do sistema implementado, com a descrição detalhada do processo de localização das mensagens enviadas na rede. Na secção 7.1 . é referida a arquitectura global do sistema implementado e referida na secção 7.2 . a nomenclatura utilizada para este sistema. Na secção 7.3 . são referidas as características do sistema implementado e na secção 7.4 . o espaço utilizado para o processo de localização. Na secção 7.5 . é descrito o processo de localização da rede de sensores implementado. Na secção 7.6 . é referida a programação e os diagramas de blocos do funcionamento dos módulos móveis e fixos. Em 7.7 . são descritas detalhadamente todos as mensagens enviadas na rede.

7.1 . Arquitectura global do sistema

A arquitectura do sistema é dividida em cinco blocos principais: rede de sensores *ZigBee, serial forwarder, middleware*, módulo de localização e aplicação de alto nível

(interface com o utilizador).

Na rede *ZigBee* são trocadas as mensagens *wireless* entre os *end devices, routers e coordinator*. Algumas das mensagens serão recebidas e/ou enviadas pelo *coordinator* da rede *ZigBee* através da placa *SmartRF04EB*, o que possibilita o reencaminhamento das mensagens para a porta série.

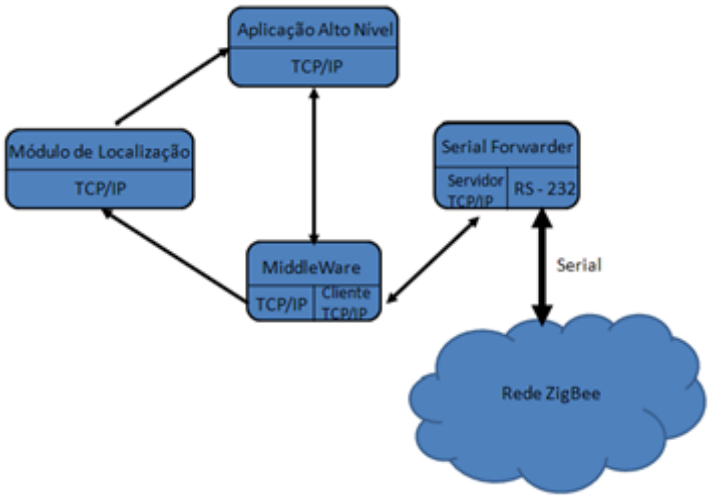

Imagem 7.1: Arquitectura global do sistema

O *Serial Forwarder* é uma aplicação que possibilita uma fácil comunicação entre a rede de sensores e o *middleware*. Esta foi previamente alterada no projecto LOPES [8] de modo a poder receber e identificar mensagens do protocolo *ZigBee*. O *Serial Forwarder* cria um servidor *TCP/IP (Transmissão Controlo Protocolo/Internet Protocolo)* onde descarrega todas as mensagens recebidas da rede de sensores, neste caso, por *RS-232*. Sempre que for recebida uma mensagem da rede de sensores (através do *coordinator* na placa *SmartRF04EB*), esta é reencaminhada para a porta série, e através do *Serial Forwarder* para um servidor *TCP*. Este servidor *TCP* transmite essas mensagem para o cliente *TCP* o que possibilita a comunicação com diversas aplicações, neste caso, uma aplicação desenvolvida em linguagem *Java*. Na direcção oposta, a mensagem é enviada do cliente *TCP* para o servidor *TCP*. Este encaminha-a para a porta série através do *Serial Forwarder* que por sua vez será enviada por *RS-232* para o *gateway SmartRF04EB*, responsável pelo envio de mensagens para a rede *ZigBee*.

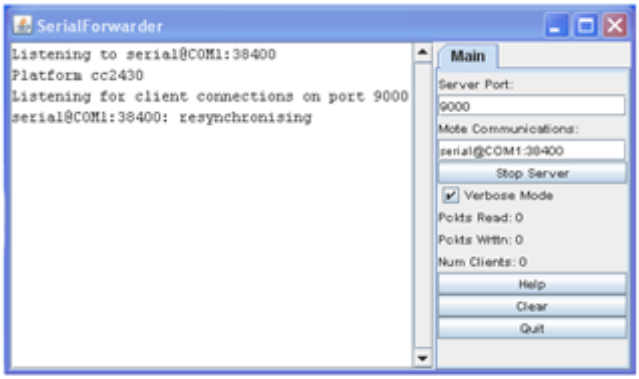

Imagem 7.2: Aplicação Serial Forwarder

O *middleware* é responsável por gerir a troca de mensagens entre a aplicação de alto nível e a rede de sensores. Permite organizar os dados das mensagens recebidas numa determinada ordem (*array* de *RSSI* ordenado) de modo a que seja possível o seu uso para a estimativa da posição no módulo de localização. Depois de organizar os dados, envia-os para o módulo de localização.

O módulo de localização recebe os dados já organizados enviados pelo *middleware* e processa-os. Deste modo são identificados os sensores móveis detectados assim como a sua posição segundo o algoritmo especificado. Em seguida, este resultado é enviado para a aplicação de alto nível que providencia a interface gráfica com o utilizador.

Esta aplicação de alto nível armazena as informações de cada sensor móvel e apresenta a sua localização num ambiente gráfico. Também é responsável pelo envio de pedidos de localização/configuração de sensores para o *middleware*.

7.2 . Nomenclatura e descrição da rede de sensores

Na dissertação considerámos a nomenclatura usada pelo *Z-location Engine da Z-Stack*, onde *BlindNode, RefNode e LocDongle* representam os *end devices, routers e coordinator* com o *firmware* específico dessa pilha protocolar.

Nome da aplicação	***Descrição***
Blind Node (BlindNode)	*End device (sensor móvel) cuja posição se deseja estimar, responsável apenas por transmissão cíclica de blasts e de adormecer entre esses ciclos*
Reference Node (RefNode)	*Router (sensor fixo de referência) que serve de suporte à localização, responsável por processamento de blasts e circulação das mensagens na rede*
Location Dongle (Loc Dongle)	*Coordinator da rede conectado a um PC através de um cabo RS-232, através da placa SmartRF04EB funciona como gateway entre a rede ZigBee e o PC*

Tabela 7.1: *Nomenclatura dos dispositivo do sistema*

7.3 . Características do sistema implementado

O sistema implementado foi programado com as seguintes características:

- um consumo energético reduzido por parte dos *BlindNodes*;
- Macro-localização de todos os *BlindNodes* baseada em proximidade, onde a posição deste será atribuída à localização do *RefNode* que detectar o seu maior nível de potência;
- detecção de um *BlindNode* específico através do uso de uma rede neuronal possibilitando uma localização mais fina;
- Configuração *wireless* dos *RefNodes* de modo a seleccionar quais os *RefNodes* que serão considerados para o processo de localização;
- detecção do primeiro *RefNode* não funcional da rede de *RefNodes* escolhida.

7.4 . Descrição do espaço de localização

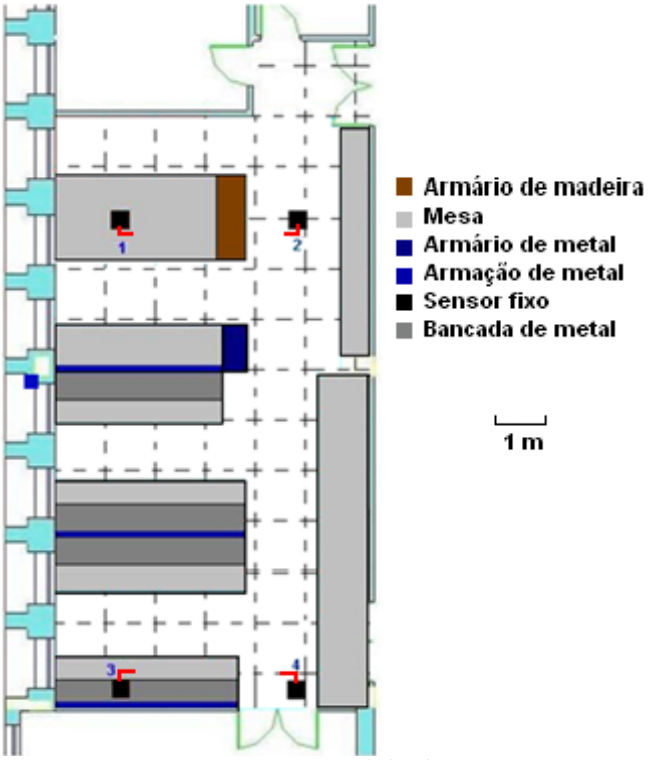

Imagem 7.3: Espaço de localização

O sistema de localização foi implementado no laboratório de Rádio-Frequência do Instituto de Telecomunicações de Aveiro representado na Imagem 7.3.

Este espaço é constituído por fontes de reflexão e refracção típicos de um sistema *indoor*. Deste modo, os resultados obtidos representam valores mais verídicos do esperado em ambientes residenciais. Os quatro sensores fixos estão montados numa estrutura de calhas técnicas junto ao tecto a uma altura de 2,60m. Para uma mais fácil ligação destes sensores fixos, a alimentação foi efectuado através de um transformador ligado em série com um interruptor que por sua vez está ligado em paralelo ao conjunto de sensores fixos. Deste modo não é necessário a utilização de pilhas AA assim como a sua troca constante.

NOTA:

Algumas ilustrações representadas em seguida sobre o sistema são referidas com 8 *RefNodes*, mas na realidade apenas foram considerados 4 como indicado na Imagem 7.3. Esta razão deve-se ao facto de uma divisão de recursos com outro projecto paralelo. A

implementação final do sistema está preparada para o funcionamento de todos os sensores da plataforma de desenvolvimento, com 8 *RefNodes*, deste modo, foram realizados os ambientes gráficos e o sistema preparados para este efeito.

7.5 . Descrição do processo de localização

Antes de iniciar a localização o *LocDongle* deverá ser ligado, seguido dos *RefNodes*. Esta ordem deve-se ao facto de qualquer rede exigir a existência de apenas um *coordinator*, assim, caso a rede seja iniciada com um *router*, este poderá assumir-se como *coordinator* (visto também ser um *FFD*) o que poderá levar a conflitos na rede.

Em seguida poderão ser ligados os *BlindNodes* que após o registo na rede iniciam um ciclo infinito onde adormecem durante um tempo pré-definido e acordam apenas para enviar uma série de mensagens, várias *LOCATIO_RSSI_BLAST* seguidos por uma mensagem identificadora do fim do ciclo, *END_OF_BLASTS_IDENTIFICATION.*

Os *RefNodes*, ao receberem os *blasts* medem o seu nível de *RSSI* e criam ou actualizam uma lista de estruturas designada por *Local_Blind_Elements.*Esta lista contém um conjunto de estruturas constituídas pelo endereço de cada *BlindNode,* um contador e os valores de *RSSI* registados pelo *RefNode* para cada *blast* recebido. Com a recepção da mensagem identificadora do fim do ciclo de *blasts* o *RefNode* calcula a média do valor de *RSSI* recebido e actualizada uma outra lista designada *BlindLocalFlowElements* constituída pelos parâmetros: média de *RSSI* e endereço do *BlindNode* correspondentes.

Através da aplicação de alto nível pode ser configurada uma rota de circulação, (*Flow_Route)*. Uma mensagem de configuração da rota *(Route_configurations)* possibilita informar todos os *RefNodes* de quais serão usados para o processo de localização e qual a rota de comunicação definida, terminando sempre no *LocDongle*.

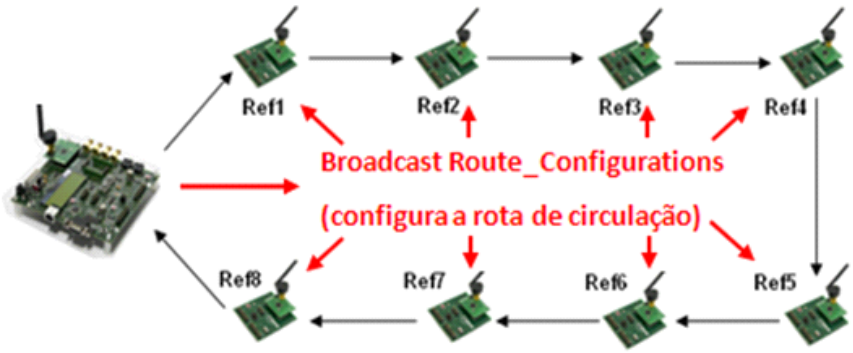

Imagem 7.4: Envio de broadcast Route_Configurations

O utilizador poderá escolher uma macro-localização baseada em proximidade onde é possível identificar todos os *BlindNodes* na área localizável. Outro modo será uma localização mais fina através de uma rede neuronal onde será localizado apenas um *Blind* previamente escolhido pelo utilizador.

No caso do objectivo ser uma macro-localização será enviado um pedido para a rede *(Hi_RSSI_FLOW_LIST)* segundo a rota de circulação especificada a requisitar a informação sobre os *Blinds* detectados, *RefNodes* que identificaram o seu maior nível de *RSSI* e esse mesmo valor. Esta mensagem irá circular pela rota especificada e actualizada em cada salto da rota. Esta actualização permite comparar e alterar o parâmetro de *RSSI* e endereço do *RefNode* mais próximo caso a detecção de *RSSI* do *RefNode* seja superior ao registado na *Hi_RSSI_FLOW_LIST* recebida. Deste modo é enviado uma mensagem para o *LocDongle* com a informação dos *BlindNodes* detectados, *RefNodes* de maior potência detectada correspondentes e *RSSI* associado como descrito na Imagem 7.5.

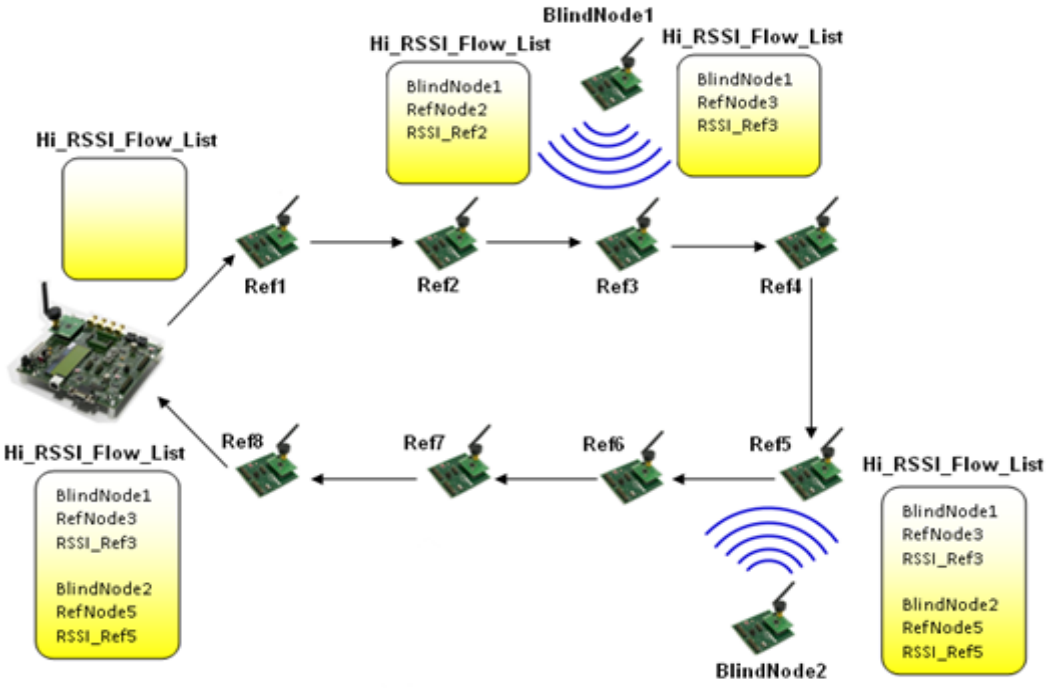

Imagem 7.5: Sistema projectado para funcionamento por proximidade

Caso o objectivo seja uma localização mais fina será enviado uma pedido para a rede *(NEURONAL_FLOW_LIST)* segundo a rota de circulação especificada a requerer os valores de *RSSI* registados pelos *RefNodes* referentes ao *BlindNode* escolhido pelo utilizador. Deste modo será recebida uma mensagem *NEURONAL_FLOW_LIST* no

LocDongle com a informação do endereço dos *RefNodes* que registaram o *BlindNode* e os seus valores de *RSSI* correspondentes. Esta mensagem depois de reencaminhada para o *middleware* será ordenada e processada pelo módulo de localização através de uma rede neuronal. Este processo é representado pela Imagem 7.6

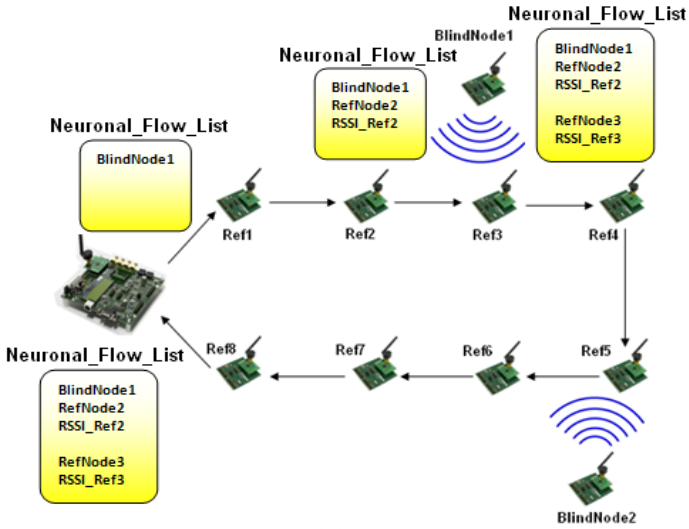

Imagem 7.6: Sistema projectado para funcionamento com rede neuronal

Se durante a rota de circulação não for retransmitido um *acknowledge* entre o nó seguinte da lista de circulação e o *RefNode* emissor, será enviado uma mensagem de identificação de falha de comunicação para o *LocDongle* com o endereço do *RefNode* não detectado. Esta mensagem será enviada indiferentemente de se tratar de uma lista de circulação *Hi_RSSI_FLOW_LIST ou NEURONAL_FLOW_LIST*. Assim é possível uma fácil identificação de problemas de comunicação entre os nós fixos. Este funcionamento é descrito na Imagem 7.7.

Imagem 7.7: Sistema projectado para alerta de falha de comunicação

7.6 . Programação dos sensores

7.6.1. Coordenador – *Loc Dongle*

O coordenador funciona como *gateway* que possibilita a comunicação entre a rede de sensores e a aplicação *middleware* que corre no servidor.

Este reencaminha as mensagens recebidas na rede de sensores sem fios para a porta série, assim como as mensagens recebidas na porta série para a rede de sensores.

A aplicação que corre no *gateway* tem o nome de *LocDongle* e vem por omissão na *Z-Stack* da *TI*. Apenas foi alterado a programação para que pudesse receber as novas mensagens criadas. Sofrendo assim, mínimas alterações de programação.

7.6.2. Sensor móvel – *Blind Node*

Quando é ligado o *Blind Node* este efectua um *Association request* à rede para saber se pode associar-se a esta. Em seguida efectua um *Data Request* ao coordenador para que lhe seja atribuído um *Short addr* válido.

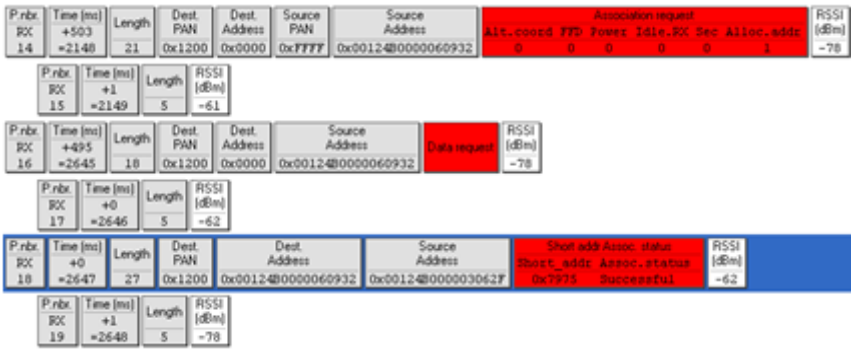

Imagem 7.8: Registo na rede de um BlindNode

No caso descrito na Imagem 7.8, o *Blind* efectua o *Association Request* e em seguida faz o pedido ao dispositivo com endereço 0x0000 (representa o coordenador) ao qual este responde informando o sucesso da associação e atribuindo-lhe o endereço 0x7975.

Após o registo na rede do *BlindNode*, este inicializa uma variável auxiliar *BLAST_CNT* de contagem com o valor de *BLINDNODE_BLAST_COUNT* e envia o primeiro *Blast (Broadcast para RefNodes sem payload com radius 1)*.

Caso o valor de *BLAST_CNT* for igual a 0 o *BlindNode* entrará num modo de

adormecimento (*SLEEP_MODE)* por um intervalo de tempo definido por *SLEEP_TIME* no ficheiro *BlindNode.c* . Caso o valor de *BLAST_CNT-1* for igual a 0, o *Blind* irá enviar uma mensagem a informar o fim de envio de *blasts* com o identificador, *ClusterID 0x0011*. Ao ser decrementado uma unidade o *BLAST_CNT* será igual a 0, o que o levará a entrar no modo *SLEEP*.

Caso o valor de *BLAST_CNT e BLAST_CNT-1* forem diferentes de 0, o valor de *BLAST_CNT* será decrementado uma unidade e um *blast* será enviado para a rede, *ClusterID 0x0019*. Este ciclo manter-se-à infinitamente não havendo mais nenhuma troca de mensagens com a rede. O funcionamento do *BlindNode* é descrito pelo diagrama de blocos representado na Imagem 7.9.

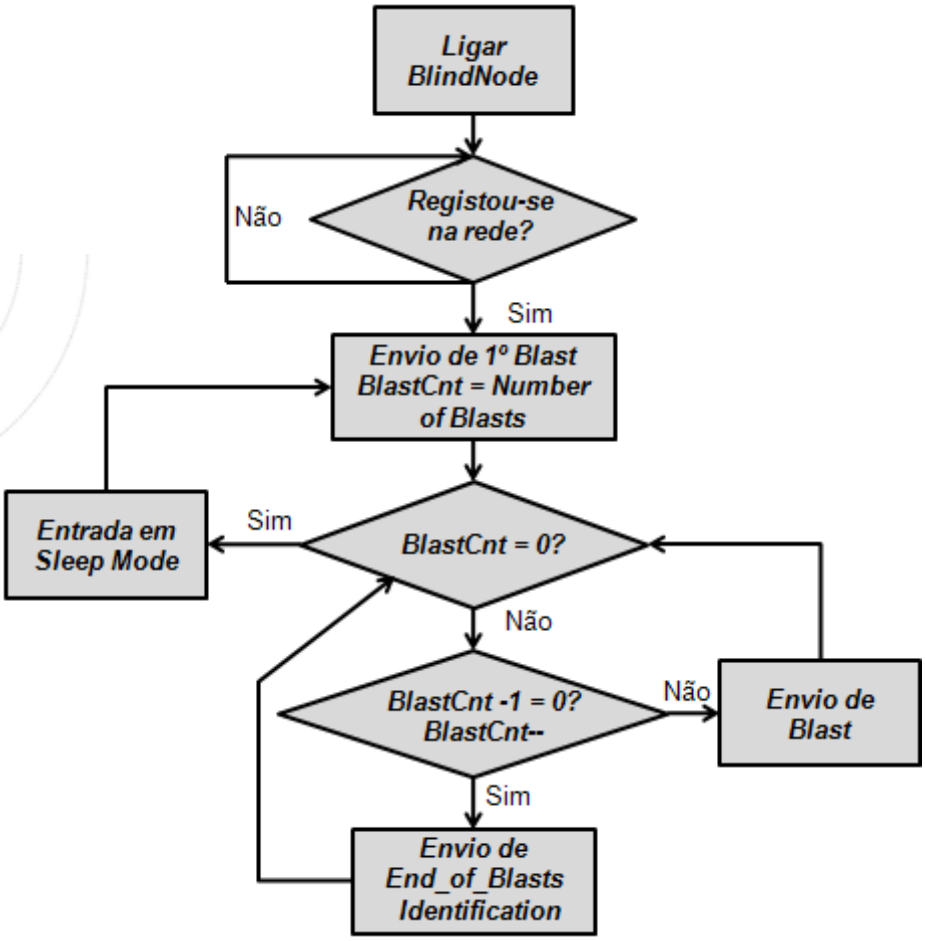

Imagem 7.9: Diagrama de blocos do funcionamento do BlindNode

Uma mensagem *blast* típica é representada pela Imagem 7.10. A informação é representada em *Little Endian* e descrita pelos seguintes elementos:

Frame Control (NPDU) – contém informações sobre o tipo de mensagem, endereçamento e outras indicações de controlo;

Destination Address – endereço destino da trama;

Source Address – endereço fonte da trama (0x2870 representa o endereço do *Blind)*;

BroadCast Radius – número de saltos máximos para o envio do *broadcast*;

BroadCast Sequence Number – número de *BroadCasts* enviados pelo *Blind*;

Frame Control (APDU) – contém informações sobre tipo de mensagem, endereçamento e outros informações de controlo;

APS dst Endpoint – especifica o *endpoint* do destinatário da mensagem (*D2-RefNodes*);

Cluster ID – representa o identificador da mensagem;

APS ProfileID – representa o identificador do *profile*;

APS Source endpoint - especifica o *endpoint* do transmissor da trama *(D3-BlindNode)*;

APS counter – identifica o número da mensagem, previne recepções duplicadas de mensagens.

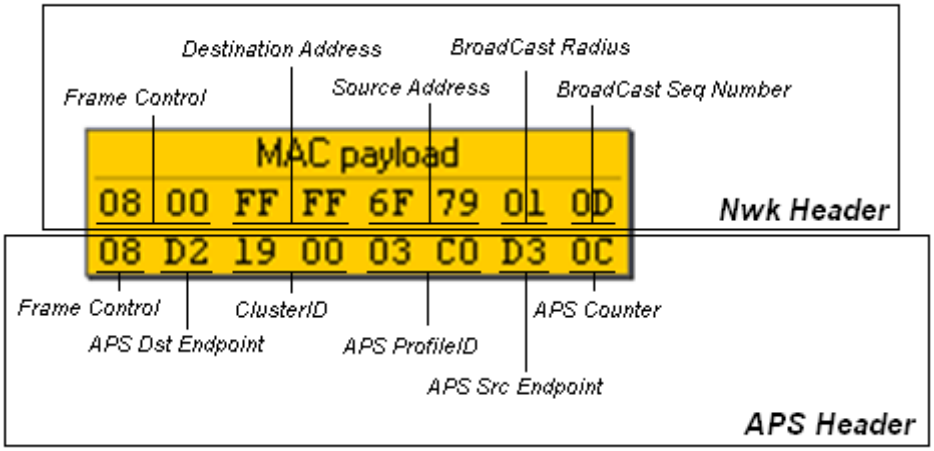

Imagem 7.10: MAC Payload (NWK Frame Format)

No ficheiro *BlindNode.c* podem ser configurados os parâmetros que vão definir o ciclo entre o envio dos *Blasts* e o ciclo temporal em que o *Blind* permanecerá em *Sleep Mode*.

A variável *SLEEP_TIME* define em milissegundos o tempo em que o *BlindNode* permanecerá em *Sleep Mode* após o envio da última mensagem do ciclo; *BLINDNODE_BLAST_DELAY* define o intervalo de tempo entre o envio de cada *blast*;

BLINDNODE_BLAST_COUNT define o número de *blasts* enviados para a rede em cada ciclo.

As imagens 7.11 e 7.12 representam um exemplo do envio de mensagens enviados pelo *Blind* e os parâmetros de configuração correspondentes.

```
// TODAS AS VARIAVEIS ENCONTRAM-SE EM MILISEGUNDOS
// tempo em que o Blind permanecerá em Sleep Mode após
// envio do ultimo Blast
#define SLEEP_TIME 5000
// delay entre envio de Blasts
#define BLINDNODE_BLAST_DELAY      20
// numero de Blasts Enviados por ciclo
#define BLINDNODE_BLAST_COUNT       8
```

Imagem 7.11: Variáveis definidas em BlindNode.c

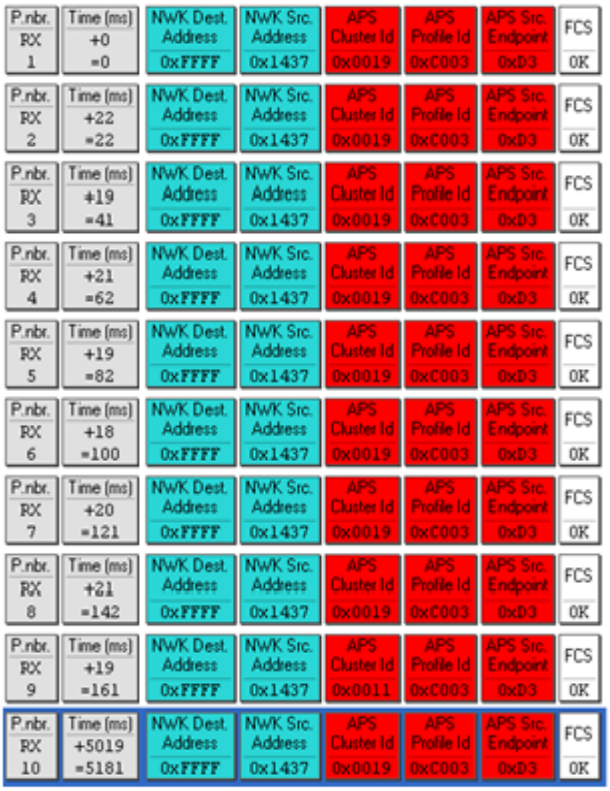

Imagem 7.12: Exemplo de ciclo de funcionamento do Blind

7.6.3. Sensor fixo - *RefNode*

O sensores fixos são os sensores que irão receber as mensagens enviadas pelos *Blinds*. Estes representam para este sistema os grandes responsáveis pelo processamento de mensagens na rede, sendo todos os outros elementos responsáveis pelo envio ou reencaminhamento de mensagens.

Após o registo na rede, os *RefNodes* permanecem constantemente à escuta de mensagens. Estas podem ter diferentes identificadores, representados na Tabela 7.2.

O diagrama de blocos representado na Imagem 7.13 identifica todo o processamento efectuado pelos *RefNodes*, onde as diferentes mensagens recebidas irão despoletar diferentes eventos, descritos em seguida.

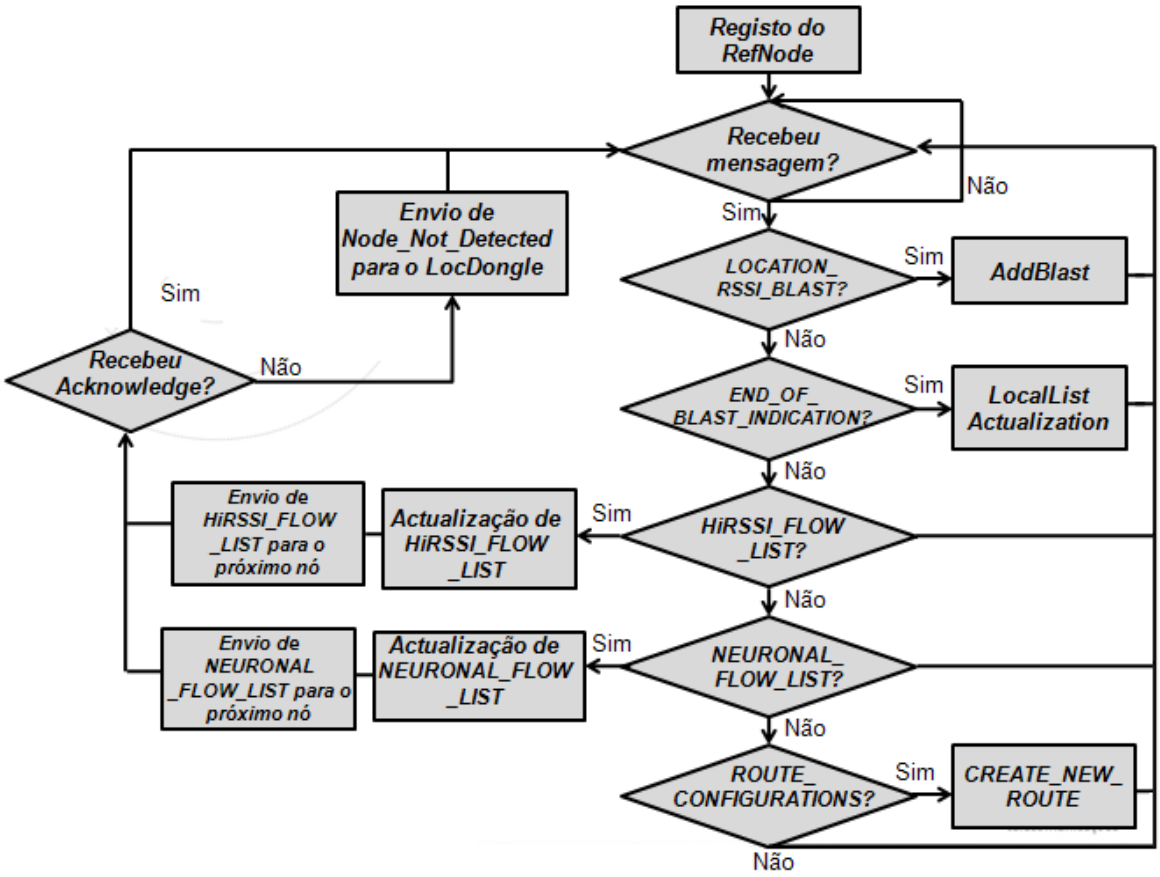

Imagem 7.13: Diagrama de blocos do funcionamento de RefNode

7.7 . Mensagens enviadas na rede e sua descrição

Cada mensagem enviada na rede tem um identificador específico, designado *ClusterID*, que possibilita ao nó que o receber despoletar eventos especificados pelo programador.

ClusterID	***Nome associado***
0x0011	*END_OF_BLASTS_IDENTIFICATION*
0x0019	*LOCATION_RSSI_BLAST*
0x0021	*TOKEN_NOT_RECEIVED*
0x0022	*HIRSSI_FLOW_LIST*
0x0023	*NEURONAL_FLOW_LIST*
0x0024	*ROUTE_CONFIGURATIONS*

Tabela 7.2: Mensagens usadas no sistema de localização

No processo de localização os diferentes dispositivos poderão receber, enviar ou ignorar as mensagens enviadas na rede. A função de cada dispositivo no modo em que interpreta as mensagens na rede é descrita na Tabela 7.3.

Mensagens	***LocDongle***	***RefNode***	***BlindNode***
0x0011	X	Recebe	Envia
0x0019	X	Recebe	Envia
0x0021	Recebe	Envia	X
0x0022	Envia/Recebe	Envia/Recebe	X
0x0023	Envia/Recebe	Envia/Recebe	X
0x0024	Envia	Recebe	X

Tabela 7.3: Interpretação das mensagens na rede por Cluster ID

7.7.1. LOCATION_*RSSI*_BLAST (0x0019)

Esta mensagem ao ser enviada pelos diferentes *BlindNodes* será recebida por todos os *RefNodes* no seu alcance. Ao serem recebidos por estes irão despoletar um processo de actualização ou criação de um novo elemento numa lista ligada de estruturas designada por *Local_Blind_Element*, representado na Imagem 7.14.

```
// Estrutura que associa cada Blind, RSSI e contagem de blasts
typedef struct _Local_Blind_Element
{
  struct _Local_Blind_Element *next;
  uint16 LocalBlindAddr;
  uint16 SumOfRSSI;
  byte cnt;
} Local_Blind_Element;
```

Imagem 7.14: Estrutura Local_Blind_Element

O valor de *RSSI* registado pelo *RefNode* será adicionado à variável *SumOfRSSI* e incrementado o valor de *cnt* (contador) ao elemento da lista que conter o *LocalBlindAddr* correspondente ao endereço do *BlindNode* que enviou a mensagem de *ClusterID 0x0019*. Caso a lista não seja constituído por nenhuma entrada com esse endereço no campo *LocalBlindAddr*, será adicionada uma nova entrada à lista com o endereço de rede do *BlindNode* em *LocalBlindAddr*, valor de *RSSI* registado em *SumOfRSSI e cnt* com o valor 1.

É de referir que o valor de *RSSI* é lido através do registo *RSSIL.RSSI_VAL* constituído por 8 bits e representado em complemento para 2. Segundo o *datasheet* do *módulo CC2431 [15]*, o valor de *RSSI* registado em *RSSI.RSSI_VAL* pode ser relacionado com a potência P nos *pins RF* pela fórmula *P = RSSI_VAL + RSSI_OFFSET [dBm]*. Esse *RSSI_OFFSET* é encontrado de um modo empírico durante o desenvolvimento do sistema de *hardware* e tem um valor de -45. Significa que por exemplo, se o registo *RSSI_VAL* indicar -20, a potência do sinal *RF* será -65dBm (-20-45 = -65dBm).

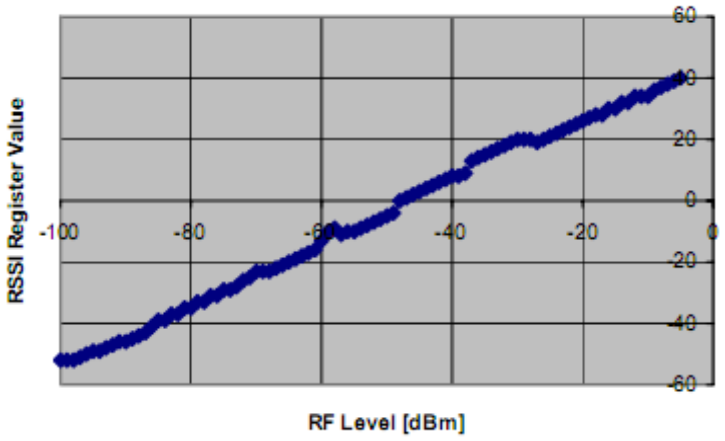

Imagem 7.15: Relação entre o valor de RSSI registado e potência RF do sinal[6]

Como a sensibilidade do *módulo CC2431* vai até -94dBm, isto significaria que o registo *RSSI_VAL* poderia ter um mínimo de -49 (-94+45). Para evitar problemas com complemento para dois e facilitar a visualização dos valores de potência reais, adicionou-se um *OFFSET* de valor +50. Deste modo todos os valores registados terão valor positivo facilitando as operações de *SumOfRSSI*. A conversão dos valores apresentados nas mensagens enviadas e os valores reais de potência são realizadas através da subtracção desse valor com a soma dos *OFFSET*s (+45+50).

7.7.2. END_OF_BLASTS_IDENTIFICATION (0x0011)

Esta mensagem enviada pelos *BlindNodes* irá despoletar o cálculo da média dos valores de *RSSI* do *Blind* correspondente. O *RefNode* através do cálculo de *SumOfRSSI/cnt* dos parâmetros da estrutura actualiza ou adiciona uma nova entrada a uma lista de estruturas designada *BlindLocalFlowElements*.

```
// Estrutura que associa BlindAddress e RSSI recebido
typedef struct _BlindLocalFlowElements
{
  struct _BlindLocalFlowElements *next;
  uint16 BlindAddress;
  byte HiRSSI;
} BlindLocalFlowElements;
```

Imagem 7.16: Estrutura BlindLocalFlowElement

Esta lista de estruturas *BlindLocalFlowElements* contém a identificação dos *Blinds* detectados pelo *RefNode* e o seu valor médio de *RSSI*. A informação do endereço do *BlindNode* é registada em *BlindAddress* e a média de *RSSI* em *HiRSSI*.

7.7.3. HIRSSI_FLOW_LIST (0x0022)

Este tipo de mensagem possibilita o processo de macro-localização. O *LocDongle* envia uma mensagem 0x0022 com *payload* nulo para o primeiro *RefNode* da lista de circulação.

Esse *RefNode* ao receber esta mensagem cria uma mensagem 0x0022 com o número de *Blinds* detectados por este, endereço desses *Blinds*, o seu endereço e o valor de *RSSI* médio registado para cada *BlindNode*. Em seguida reencaminha esta mensagem para o *RefNod*e seguinte da rota de circulação que irá efectuar o mesmo processo. O número

máximo de *blinds* detectáveis por este modo é de 20 (103*bytes (max payload - 1)*/5).

A mensagem representada na Imagem 7.17 indica a detecção de *3 Blinds pelo RefNode* de endereço 0x0001. Como referido anteriormente, o *OFFSET* total é de +45+50, deste modo, o *Blind* de endereço 0x7971 apresenta *RSSI* (4 – *OFFSET* = -91); o *Blind* 0x7970 de *RSSI* (6 – *OFFSET* = -89); e o *Blind* 0x7972 de *RSSI* (21 – *OFFSET* = -74).

Esta mensagem irá circular por uma rota previamente definida e será actualizada de acordo com novos *Blinds* detectados e os *RefNodes* que detectarem maiores níveis de *RSSI*. Sendo sempre terminada no *LocDongle*.

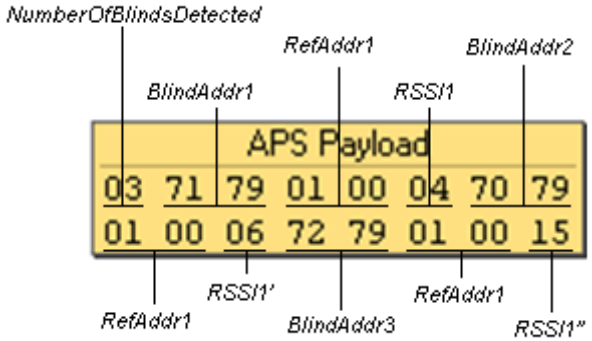

Imagem 7.17: Mensagem HIRSSI_FLOW_LIST enviada pelo 1º RefNode

Na mensagem da Imagem 7.18 é representada uma mensagem após passar por vários *RefNodes*. O número de *Blinds* detectados na rede manteve-se mas os *RefNodes* a identificar o maior valor de potência foi alterado.

Por exemplo, o *Blind* 0x7971 foi registado com maior valor de potência pelo *RefNode* 0x143E com o valor de (24 – *OFFSET* = -71) enquanto o *Blind* 0x7972 foi registado com o maior valor de potência sempre pelo mesmo *RefNode* (0x0001) com o valor de (21 – *OFFSET* = -74).

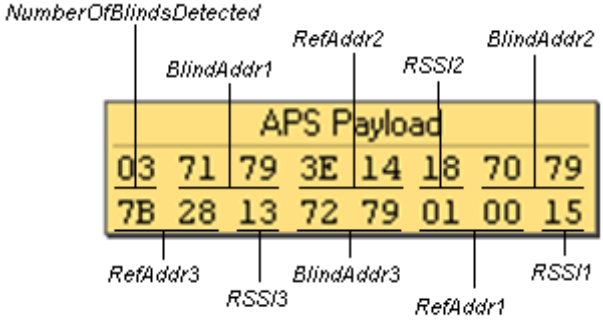

Imagem 7.18: Mensagem HIRSSI_FLOW_LIST enviada pelo 3º RefNode

7.7.4. NEURONAL_FLOW_LIST (0x0023)

Este tipo de mensagem é responsável pela recolha de dados necessários para o processo de localização através de uma rede neuronal. O *LocDongle* irá enviar uma mensagem *NEURONAL_FLOW_LIST* a indicar o endereço do *Blind* que se deseja localizar seguido de um *byte* (iniciado a 0 pelo *LocDongle*) a indicar o número de *RefNodes* que detectarem esse *Blind.* No exemplo da Imagem 7.19 o endereço do *Blind* que se pretende localizar tem endereço 0x7972. A mensagem será enviada e actualizada segundo a rota de circulação indicando o *BlindAddress*, o número de *RefNodes* que detectaram o *BlindNode*, o endereço dos *RefNode Address* e os valores de *RSSI* registados por estes.

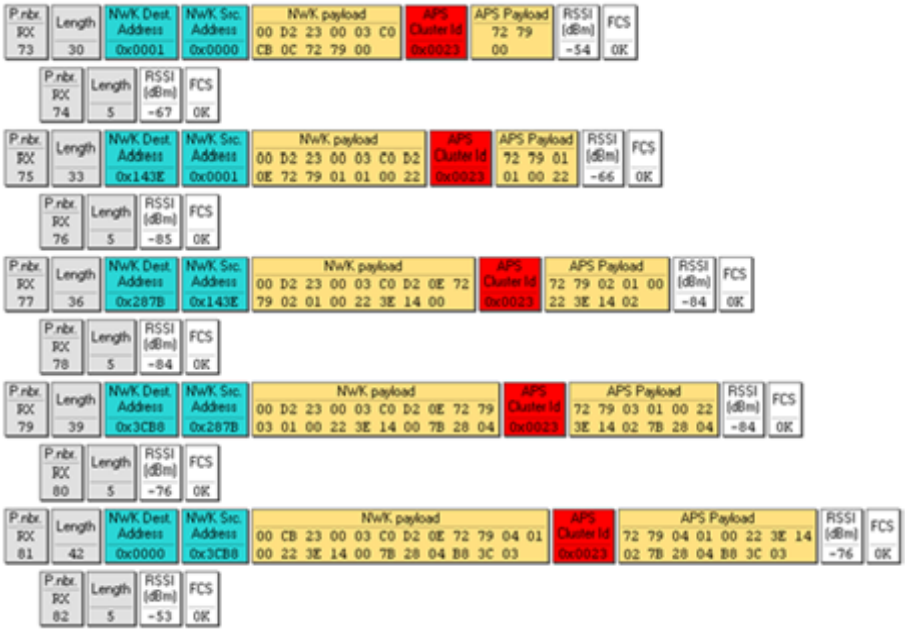

Imagem 7.19: Exemplo de NEURONAL_FLOW_LIST

É de referir que na Imagem 7.19 o envio da *NEURONAL_FLOW_LIST* é iniciada no endereço 0x0000, referente ao *LocDongle,* sendo reenviada e terminada com o envio para esse mesmo endereço, proveniente do último *RefNode* de endereço 0x3CB8.

Na Imagem 7.20 é descrito de um modo mais detalhado o significado de cada campo da mensagem *NEURONAL_FLOW_LIST.*

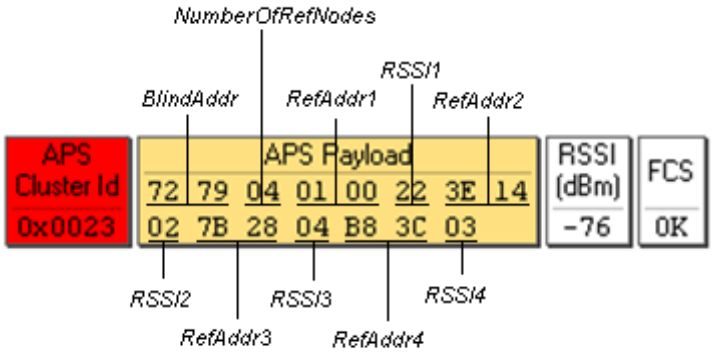

Imagem 7.20: Descrição de uma mensagem NEURONAL_FLOW_LIST

7.7.5. ROUTE_CONFIGURATIONS (0x0024)

Esta mensagem *broadcast* é enviada pelo *Coordinator* para todos os *RefNodes* especificando quais os *RefNodes* que constituem a rota de circulação, *FLOW_ROUTE*, que será considerada. Na Imagem 7.21 a rota de circulação considerada foi: *RefNode* de endereço 0x0001, 0x143E,0x0005 e 0x3CB8.

Esta mensagem configura todos os *RefNodes* indicando a *FLOW_ROUTE* que será considerada no processo de captura de valores de *RSSI*. O processo de localização seria então iniciado no *LocDongle*, transmitido para o *RefNode* de endereço 0x0001; retransmitido para o *RefNode* de endereço 0x143E; em seguida para o *RefNode* 0x0005 e terminando no *RefNode* 0x3CB8. Este *RefNode*, sendo o último elemento da *FLOW_ROUTE* enviará a mensagem do processo de localização para o *LocDongle* de endereço 0x0000.

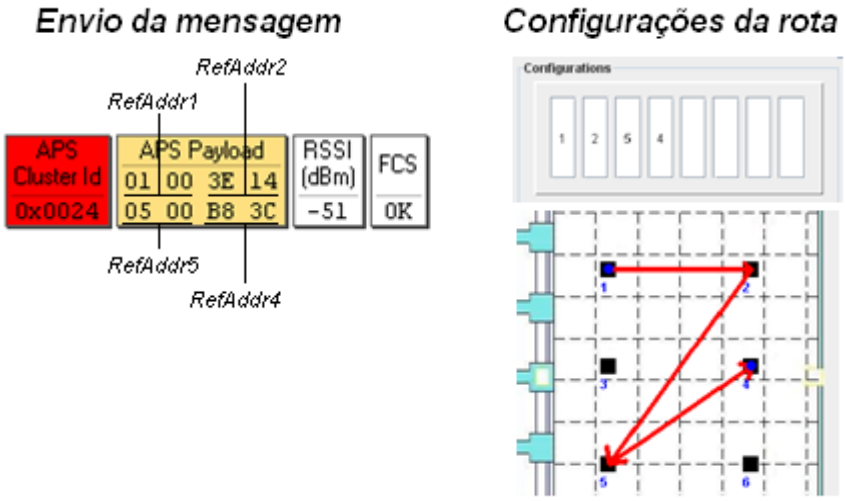

Imagem 7.21: Exemplo de uma mensagem FLOW_ROUTE

7.7.6. TOKEN_NOT_RECEIVED (0x0021)

Esta mensagem é enviada para o coordenador a informar a falha de comunicação com um nó da *FLOW_ROUTE*. Na Imagem 7.22 é representado uma tentativa de comunicação do *RefNode* 0x143E com o dispositivo 0x287B (nó seguinte da *FLOW_ROUTE*). Após a tentativa de comunicação e não recepção de um *acknowlegde*, este dispositivo envia uma mensagem de *ClusterID* 0x0021 para o *LocDongle* (0x0000) com a informação do endereço do dispositivo que não enviou o *acknowlegde*, o próximo nó da rota de circulação. Neste caso o dispositivo 0x143E envia uma mensagem com *APS payload* a indicar o endereço do dispositivo 0x287B para o *LocDongle* de endereço 0x0000.

P.nbr.	Dest. Address	Source Address	APS Cluster Id	APS Payload	RSSI (dBm)	FCS
RX 15	0x287B	0x143E	0x0022	01 E9 50 01 00 17	-85	OK
RX 16	0x287B	0x143E	0x0022	01 E9 50 01 00 17	-86	OK
RX 17	0x287B	0x143E	0x0022	01 E9 50 01 00 17	-87	OK
RX 18	0x287B	0x143E	0x0022	01 E9 50 01 00 17	-85	OK
RX 19	0x0000	0x143E	0x0021	7B 28	-85	OK

Imagem 7.22: Exemplo de uma mensagem de TOKEN_NOT_RECEIVED

Capítulo 8

Interface com o utilizador

De modo a permitir uma fácil utilização do sistema de localização foi desenvolvida uma interface gráfica de fácil utilização em linguagem *Java*. Esta é dividida em duas aplicações: uma aplicação para calibração (fase *offline*), onde é feita a recolha de assinaturas *RF* e fase *online* que permite o processo de localização dos *BlindNodes* baseada em proximidade ou numa rede neuronal conforme o pretendido pelo utilizador.

8.1 . Fase de calibração (fase offline)

Neste modo é efectuada a recolha de *"fingerprints"*, conjunto de potências registadas por cada *RefNode* na detecção de um *BlindNode* numa determinada posição.

O funcionamento desta aplicação é bastante simples e representado na Imagem 8.1. Após executar esta aplicação, *"Data Collection Menu"* deve-se clicar no botão *Find Blinds*, representado por (4). Este procedimento irá enviar uma mensagem 0x0022 para a rede o que permitirá localizar todos os *BlindNodes* na recepção desta mesma. A identificação dos *BlindNodes* deverá aparecer na caixa de texto (5) identificando os *BlindNodes* através do

seu endereço de rede *(short address)* em valor decimal. Selecciona-se o *BlindNode* que se pretende calibrar em (5) o que deverá automaticamente inserir esse endereço em (7). Escreve-se o número de identificação da posição no campo (1), selecciona-se a posição no mapa onde se encontra o *BlindNode*, com o rato e inserimos a altura no campo *Height* (2). Deverá aparecer uma posição a três dimensões no campo (3). Em seguida clica-se no botão *Start Data Collection* (6). Este processo irá despoletar o envio de uma mensagem *NEURONAL_FLOW_LIST* (0x0023) para a rede permitindo a quando da sua recepção a criação de um ficheiro *.dat*. O nome destes ficheiros será o inserido no campo *Position ID* (1) e terá como conteúdos a posição do campo (3) e o conjunto dos valores ordenados de potência recebidos.

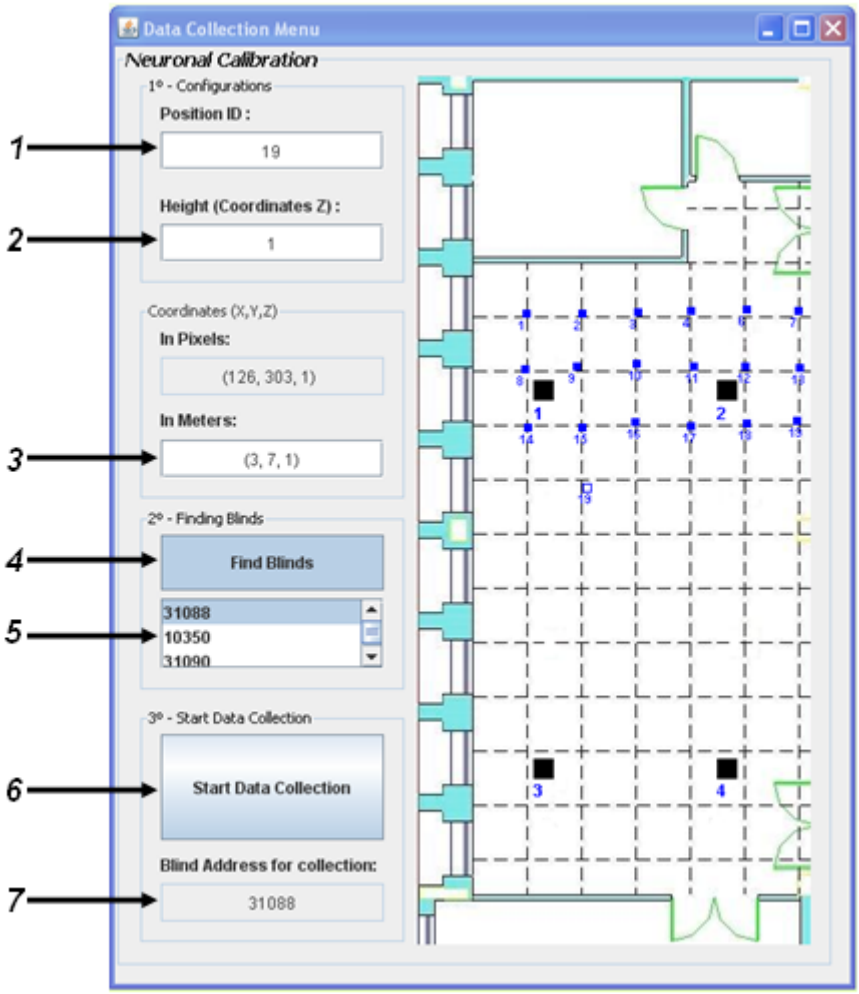

Imagem 8.1: Data Collection Menu

Depois de fazer a calibração de toda a rede ter-se-à que criar um ficheiro com todos os dados da calibração e inserir na primeira linha os parâmetros de entrada para o executável que irá criar a rede neuronal (*simple_train.exe*). Esta linha terá o número total de amostras, o número de médias de *RSSI* (um por cada *RefNode*) e o número de coordenadas de saída (3 – correspondentes a X,Y,Z). A Imagem 8.2 representa um excerto desse ficheiro. Contém 132 pontos de calibração, cada um deles com 4 parâmetros de entrada e 3 de saída.

```
132 4 3
-0.6727272727272727 -0.8181818181818181 -0.6363636363636364 -0.49090909090909096
2 10 1
-0.5636363636363637 -0.6727272727272727 -0.4181818181818182 -0.49090909090909096
2 10 2
-0.23636363636363633 -0.6 -0.6363636363636364 -0.6727272727272727
2 11 1
-0.5636363636363637 -0.6 -0.8181818181818181 -0.34545454545454546
2 11 2
-0.5272727272727273 -0.7090909090909091 -0.8181818181818181 -0.6727272727272727
2 12 1
```

Imagem 8.2: Ficheiro de calibração para a rede neuronal

Este ficheiro deverá ser guardado no directório raiz do programa que irá criar a rede neuronal para que possa ser executado. O programa utilizado foi o *Fast Artificial Neural Network Library (FANN)* [44] que contém o executável *simple_train.exe* para criar a rede neuronal e o executável *simple_test.exe* para processar a rede neuronal criada. Ao executar este programa deverá ser criado o ficheiro NEURONAL_NETWORK_IT.*net* no mesmo directório onde foi inserido o ficheiro com a calibração da rede. No caso do sistema implementado seria em *D:\JavaProjects\PlantaIT\FastFannNeuralNetwork\fann-2.0.0\MicrosoftVisualC++.Net.*

Este ficheiro em seguida deverá ser colocado no directório raiz da aplicação *Java* desenvolvida para que possa ser acedido na fase *online* através do executável *simple_test.exe.* No projecto desenvolvido o directório onde deverá ser colocado esse ficheiro é *D:\JavaProjects\PlantaIT*.

8.2 . Fase online

8.2.1. Main Menu

Este menu de simples utilização permite ao utilizador efectuar o processo de localização sem que necessite de um conhecimento prévio sobre sistemas ou algoritmos de localização. O menu é representado pela Imagem 8.3.

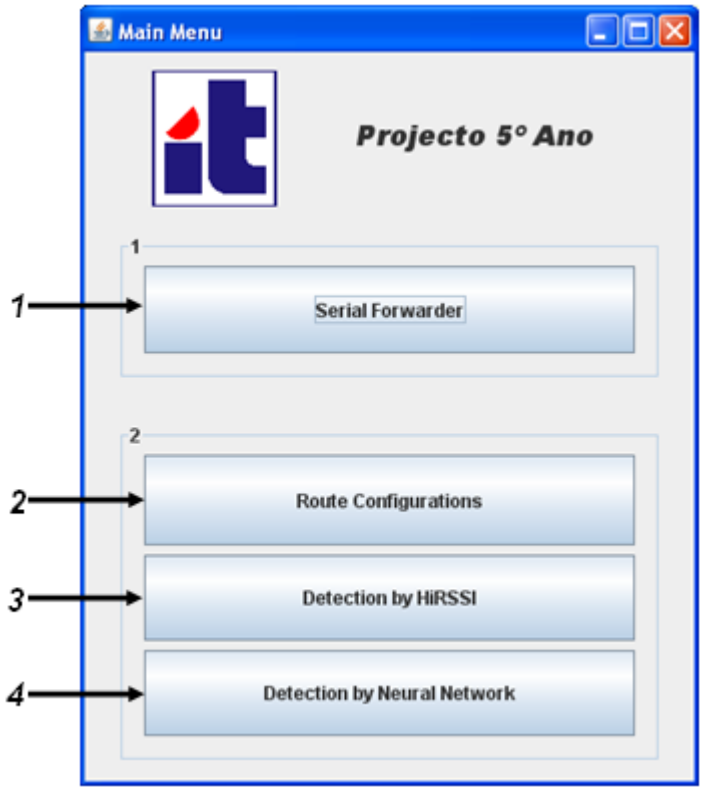

Imagem 8.3: Main Menu

O processo de comunicação deverá ser iniciado sempre com o *Serial Forwarder (1)* de modo a estabelecer a comunicação por *RS-232* com o *gateway*. Em seguida o utilizador é livre de utilizar qualquer uma das opções descrita no menu conforme o objectivo pretendido. Poderá realizar a configuração da rota das mensagens através do botão *"Route Configurations"(2);* detecção de todos os *BlindNodes* na rede por proximidade através do botão *"Detection by HiRSSI"(3)*; ou detecção de um *BlindNode* específico através de uma rede neuronal através do botão *"Detection by Neuronal Network"(4)*.

8.2.2. Serial Forwarder

O primeiro passo a executar na aplicação é o *Serial Forwarder* como descrito anteriormente, o que permite a comunicação da aplicação *Java* desenvolvida com a rede de sensores *ZigBee* através de uma porta série. Neste processo ter-se-à que seleccionar a porta série *(2)* e o porto *TCP/IP (1)* apropriados. O Porto *TCP* tem que ser o número 9000 visto ser o considerado na aplicação, a porta série dependerá da porta série de comunicação com o *gateway*. Em seguida deverá ser pressionado *Stop Server e Start Server* (3) de modo a iniciar a comunicação. Na caixa de texto deverá aparecer a mensagem "*resynchronising*", indicando uma correcta comunicação entre o porto *TCP* e a porta série.

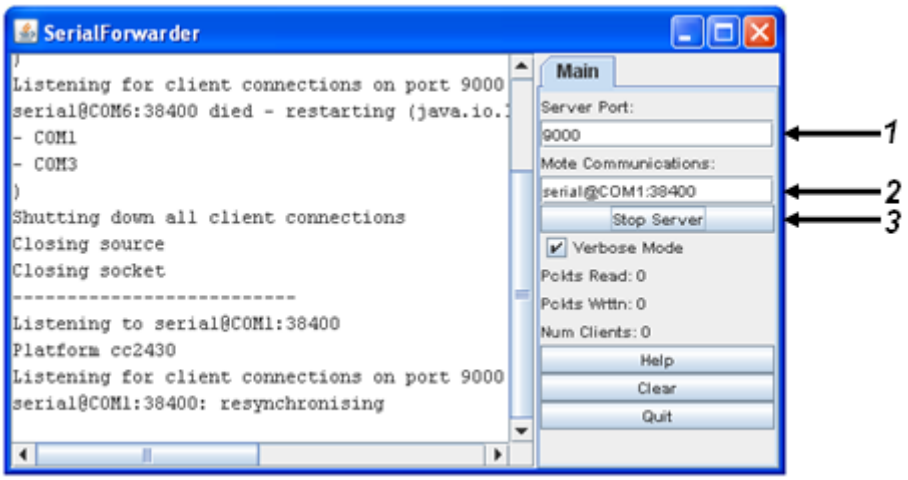

Imagem 8.4: Descrição do Serial Forwarder

8.2.3. Route Configurations

Este menu permite a configuração dos *RefNodes* que irão constituir a lista de circulação da rede para localização de *BlindNodes* representado pela Imagem 8.5.

O utilizador deverá inserir a identificação dos *RefNodes*, neste caso de 1 a 8 no campo (2) conforme a rota que pretender realizar, caso pretenda voltar à rota *default* deverá clicar no botão representado pelo campo (5). Em seguida deverá clicar no botão "*Preview*" o que possibilitará visualizar o mapa da rota seleccionada. Se pretendermos inserir novos valores poderemos sempre limpar todos os campos através do botão "*Reset*" representado no campo (4).

Após obtermos a visualização dos *RefNodes* que constituirão a lista de circulação dever-se-à clicar no botão "*Send Message*" representado no campo (1).

Uma mensagem *ROUTE_CONFIGURATIONS* será enviada para a rede o que permitirá a configuração dos *RefNodes*.

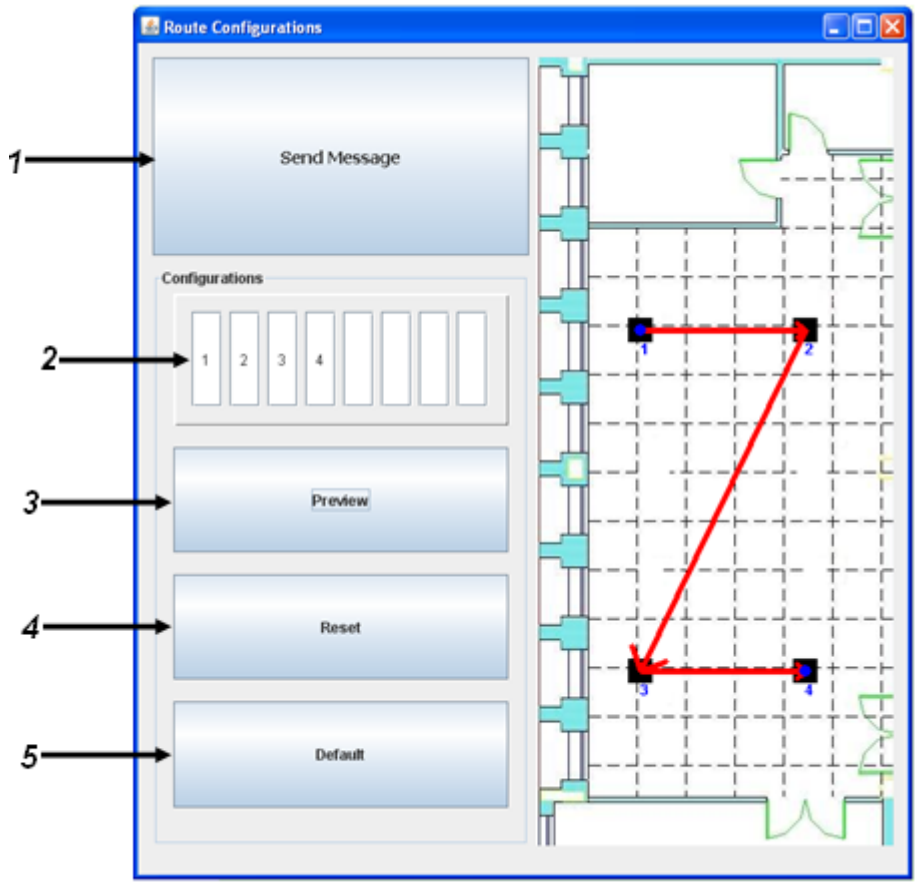

Imagem 8.5: Route Configurationd Menu

8.2.4. Detection by HiRSSI

Esta aplicação permite a localização de todos os *BlindNodes* na rede, identificando cada um deles com uma cor e número segundo a ordem de detecção na rede.

Se pretendermos alterar a duração do ciclo de actualização da detecção basta alterar o campo *(2)* para o ciclo (em segundos) que se pretender. A localização terá início quando o utilizador clicar no botão "*Start*" representado pelo campo *(1)*. Isto irá despoletar um envio periódico de mensagens *HIRSSI_FLOW_LIST* para a rede o que possibilitará a detecção baseada no maior nível de potência detectada pelos *RefNodes* da lista de circulação. Ao receber a mensagem *HIRSSI_FLOW_LIST* automaticamente serão apresentados a identificação dos *BlindNodes* no campo *(5)* e a sua posição no mapa previamente criado. No campo *(5)* são representados o *BlindNode Address*, cor e número

associados a cada *BlindNode,* estando o número e cor associados à ordem em que estes foram localizados na aplicação.

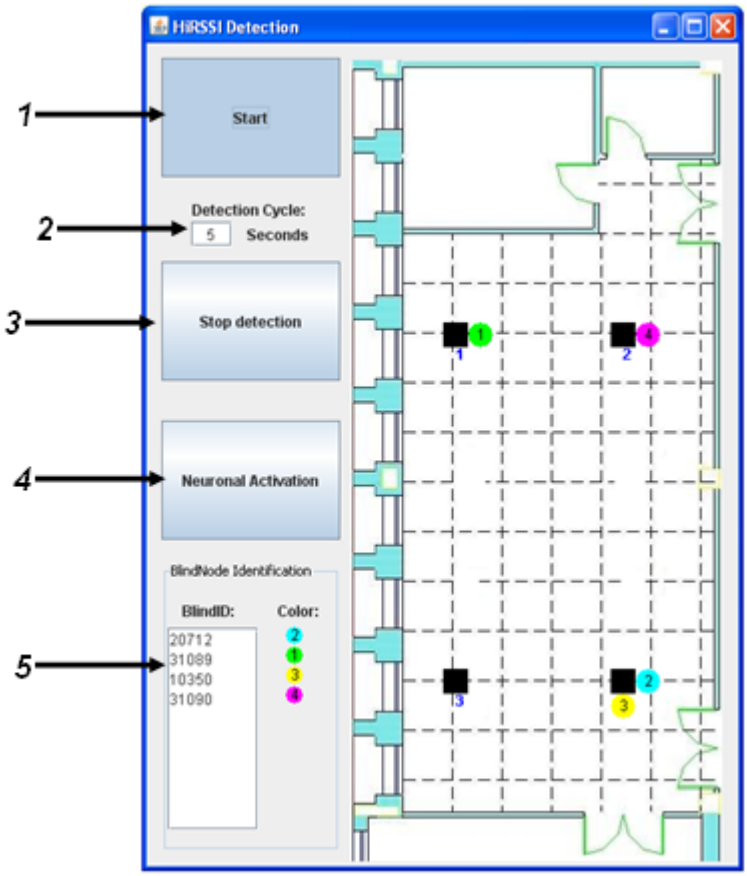

Imagem 8.6: HiRSSI Detection Menu

É de referir que a posição apresentada no mapa de localização apenas está associada com o maior nível de potência sendo a representação do *BlindNode* na posição superior, inferior, esquerda ou direita não relacionada com uma posição mais precisa mas sim devido a um número crescente de *BlindNodes* detectados por esse *RefNode*.

O ciclo poderá ser interrompido através do botão "*Stop detection*" referido pelo campo *(3)*. O campo *(4)* possibilita uma entrada directa para o menu de localização através de uma rede neuronal.

8.2.5. Detecção através da rede Neuronal

Esta aplicação possibilita a detecção de um *BlindNode* através da rede neuronal.

Para o funcionamento desta aplicação dever-se-à clicar no botão referido pelo campo *(4)* o que possibilitará visualizar os *BlindNodes* detectados no campo *(5)*. Selecciona-se em

seguida o *BlindNode* pretendido (em 5) e o ciclo de actualização da detecção *(2)*. A partir deste momento é possível iniciar o processo de localização através do botão *"Start"(1)*.

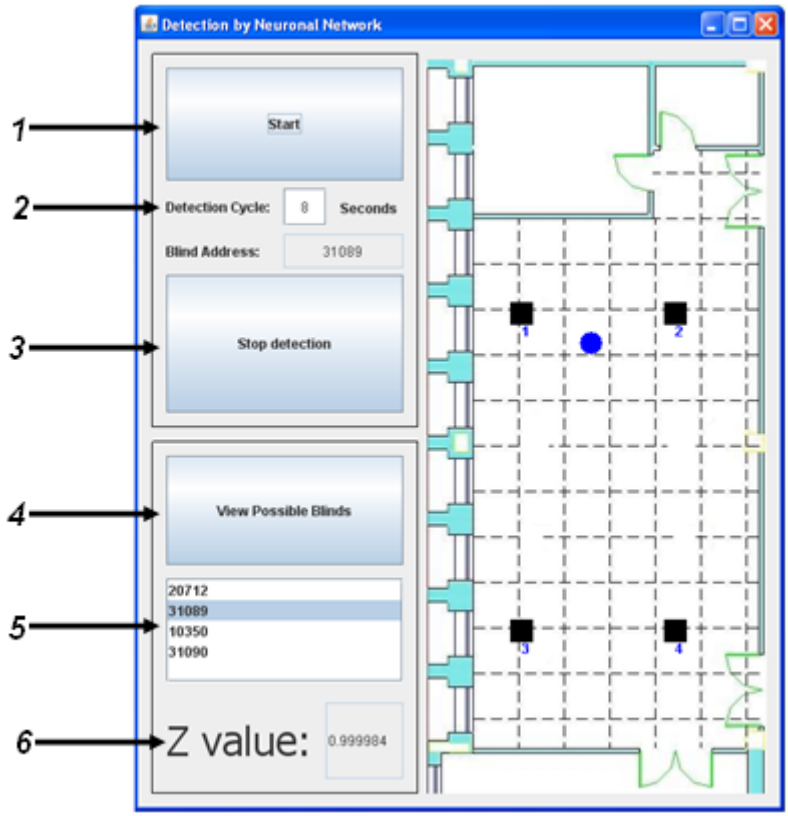

Imagem 8.7: Detection by Neuronal Network Menu

Este processo irá despoletar o envio de uma mensagem *NEURONAL_FLOW_LIST* para a rede. Após a recepção desta mensagem pelo *LocDongle* será enviada para o *middleware*. Após a ordenação dos valores de *RSSI* conforme a ordem usada na calibração da rede neuronal, este parâmetro será inserido e processado pela rede neuronal através do executável *simple_test.exe* que irá comparar o vector de *RSSI* recebido com a rede neuronal *NEURONAL_NETWORK_IT.net*, determinando assim a posição do *BlindNode*. De acordo com o parâmetro de saída da rede neuronal será visualizada a posição a duas dimensões no mapa e o valor da altura na caixa de texto *Z-value(6)*.

É de mencionar que o executável está preparado para funcionar apenas caso o número de parâmetros de entrada seja igual ao número de parâmetros considerados na calibração da rede neuronal. Para evitar problemas devida a uma não identificação por parte de um ou mais *RefNodes*, o valor de potência considerado para *RefNodes* que não detectarem o *BlindNode* será considerado como o menor valor de potência detectado na

mensagem recebida.

```
const unsigned int num_input = 4;
const unsigned int num_output = 3;
const unsigned int num_layers = 3;
const unsigned int num_neurons_hidden = 10;
const float desired_error = (const float) 0.00001;
const unsigned int max_epochs = 500000;
const unsigned int epochs_between_reports = 10000;
```

Imagem 8.8: Parâmetros considerados para a rede neuronal

O programa *Simple_train.exe* cria uma rede neuronal com os parâmetros representados na Imagem 8.8. Como está indicado na imagem, foram considerados 4 parâmetros de entrada associados aos 4 *RefNodes* considerados para o processo de localização, 3 parâmetros de saída correspondentes à posição a três dimensões e 3 camadas ocultas com 10 neurónios cada.

8.2.6. Erro de comunicação na rede

Caso o envio da lista de circulação seja interrompido e não se verifique um *acknowledge* no envio da lista de circulação na rede entre o nó emissor e o nó seguinte, será enviada uma mensagem de *TOKEN_NOT_RECEIVED* para o *LocDongle* identificando assim qual o *RefNode* não funcional. Na aplicação *Java* será identificada essa mensagem e visualizado um quadrado vermelho na posição do *RefNode* que não recebeu a *FLOW_LIST* como demonstrada na Imagem 8.9.

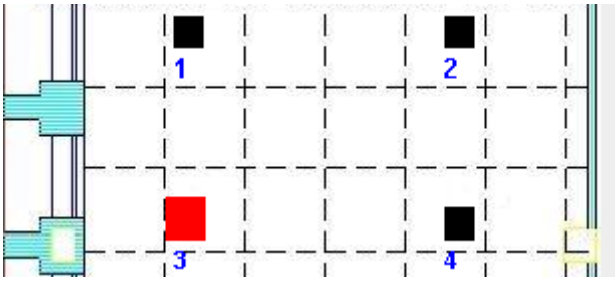

Imagem 8.9: Identificação de erro de comunicação

Capítulo 9

Resultados

Neste capítulo são descritos os resultados do consumo energético dos módulos móveis conforme diferentes baterias e uma comparação com o consumo energético no sistema inicial em que este projecto se baseou. Em seguida serão descritos os resultados em termos de exactidão e precisão do sistema de localização de acordo com as antenas fornecidas pelo *kit* de desenvolvimento e com as antenas sugeridas e criadas neste projecto *(microstrip)*. Na primeira secção será descrito o modo como foi efectuado a medição do consumo energético do *BlindNode e* uma descrição detalhada do seu funcionamento. É também referido o cálculo da duração de vida esperada segundo várias baterias. Na segunda secção serão apresentados os resultados de exactidão e precisão sobre o processo de localização.

9.1 . Consumo energético dos módulos móveis

9.1.1. Considerações iniciais

Para diminuir o consumo energético dos *BlindNodes* ao máximo, estes foram configurados para funcionar em *AUTO-MODE*, isto significa que após o registo na rede não responderão a *requests*, tornando as suas acções independentes da rede de sensores[15].

Ao ligar o *EndDevice* este fará um pedido inicial para se ligar à rede adquirindo assim um endereço de rede. Após este pedido terá um funcionamento independente do sistema, realizando ciclicamente o envio de *N Blasts* seguidos de X segundos em *Sleep Mode (Lite Sleep ou PM2).*

A *Z-Stack* possibilita dois *Sleep Modes*, designados *LITE e DEEP Sleep*, sendo o primeiro usado quando se pretende acordar o *BlindNode* através de um *timer*, enquanto o segundo é usada quando nenhuma tarefa está agendada, sendo forçado o acordar através de uma interrupção externa.

Para o sistema pretendido interessa-nos o funcionamento em *LITE Sleep*.

Os requisitos para entrar em *Sleep Mode são:*

1. A opção de compilação *POWER_SAVING* tem que estar activa;
2. A recepção (*Rx)* tem que estar desligada quando desocupada, deverá ser alterado a configuração *RFD_RCVC_ALWAYS_ON* para falso no ficheiro *f8wConfig.cfg*;
3. Todas as tarefas da *Z-Stack* têm que terminar em *Power Saving*;
4. Não poderá haver tarefas agendadas na *Z-Stack* na camada *MAC*;

Como o sistema não necessita de receber mensagens foram configurados todos os *poll rates* a 0, P*OLL_RATE, QUEUED_POLL_RATE e RESPONSE_POLL_RATE*.

9.1.2. Sistema de medição

Para a análise dos consumos energéticos dos *BlindNodes* foi realizado um circuito eléctrico básico de modo a medirmos a corrente de entrada do módulo móvel. Foi ligado em série o módulo *CC2431* com uma resistência de *10 Ohms*. Através da análise da queda de tensão na resistência com um osciloscópio digital é possível estimar a corrente consumida pelo módulo, e através desta e da análise temporal é possível calcular o

consumo deste dispositivo.

Esse sistema é representado pela Imagem 9.1

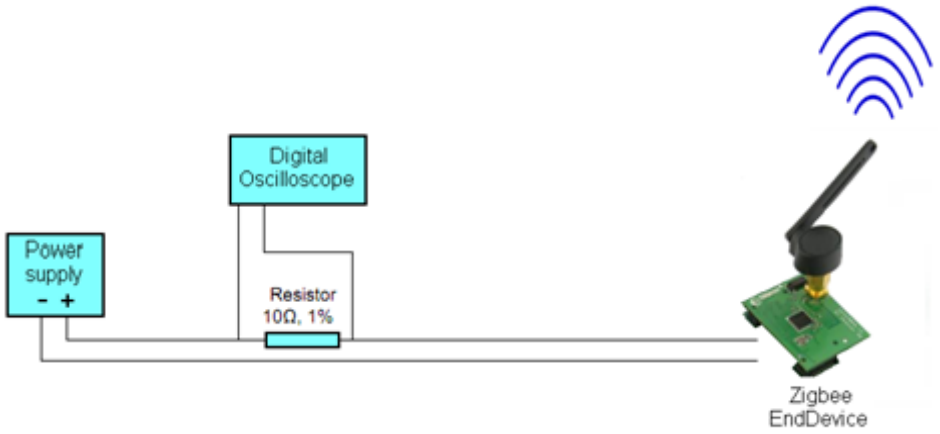

Imagem 9.1: Sistema de medição de corrente

A ligação da fonte de tensão ao módulo *CC2430EM* é feita através da ligação de:

- *GND a P2 pin 2;*
- *VDD +3.0V a P2 pin 9;*

Para a medição do consumo energético do módulo foi considerado o caso limite de *BLINDNODE_BLAST_DELAY = 0* e um *BLINDNODE_BLAST_COUNT = 5*.

A análise entre o *sniffer* e o osciloscópio referem alguns desfasamentos temporais o que é normal, devido a atrasos de propagação o que se pode verificar nas diferenças temporais entre as imagens 9.2 e 9.3.

P.nbr.	Time (ms)	APS	RSSI	FCS
RX 1	+0 =0	Cluster Id 0x0019	(dBm) -43	OK
RX 2	+5 =5	Cluster Id 0x0019	(dBm) -44	OK
RX 3	+6 =11	Cluster Id 0x0019	(dBm) -43	OK
RX 4	+5 =16	Cluster Id 0x0019	(dBm) -43	OK
RX 5	+5 =22	Cluster Id 0x0019	(dBm) -43	OK
RX 6	+6 =28	Cluster Id 0x0011	(dBm) -43	OK
RX 7	+5003 =5032	Cluster Id 0x0019	(dBm) -43	OK

Imagem 9.2: Análise do ciclo de mensagens do Blind através do Sniffer

9.1.3. Medição do consumo num ciclo típico

O consumo energético dos módulos *CC2430/31* está directamente relacionado com quatro modos de funcionamento *(PM0 a PM3)* possíveis [15], descritos na Tabela 9.1.

O modo *PM0*, também designado por modo activo, permite aceder a todas as funcionalidades de operação do módulo, onde *CPU*, periféricos e transceptores estão activos.

O *PM1* é usado quando o intervalo de tempo entre adormecimento e o acordar do módulo é inferior a 3ms, isto porque permite um transição mais rápida para o *PM0* comparado com o *PM2*. É referido um consumo de 190μA e uma transição para *PM0* de 4.1μs.

O *PM2* é usado quando se pretende acordar o dispositivo através de um timer e/ou interrupções externas. É referido um consumo energético de 0.5μA e uma transição para *PM0* de 120μs. Este é usado quando o tempo de *Sleep* excede 3ms e pode ir até um tempo máximo *Sleep Time* de 510 segundos.

O *PM3* é usado quando apenas uma interrupção externa poderá despoletar o acordar do módulo. Este modo permite um consumo energético de 0.3μA e uma transição igual ao *PM2*, 120μs.

Para o sistema usado são usados os modos *PM0 e PM2*.

Power Mode	High-frequency oscillator	Low-frequency oscillator	Voltage regulator (digital)
Configuration	A None B 32 MHz XOSC C 16 MHz RCOSC	A None B 32.753 kHz RCOSC C 32.768 kHz XOSC	
PM0	B, C	B or C	ON
PM1	A	B or C	ON
PM2	A	B or C	OFF
PM3	A	A	OFF

Tabela 9.1: Modos de funcionamento para o módulo cc2430/31 [15]

Na Imagem 9.3 é representada uma análise do consumo energético dos *BlindNodes* visualizado com um osciloscópio digital. As mensagens e fases do ciclo do *BlindNode* são descritas pormenorizadamente na Tabela 9.2. Pode-se verificar o envio de 6 mensagens, correspondendo ao envio de 5 *blasts* seguidos de uma mensagem da identificação do fim de envio de *blasts*. Este envio é feito em *PM0* seguido-se uma intervalo de cinco segundos no estado de *Sleep Lite (PM2).*

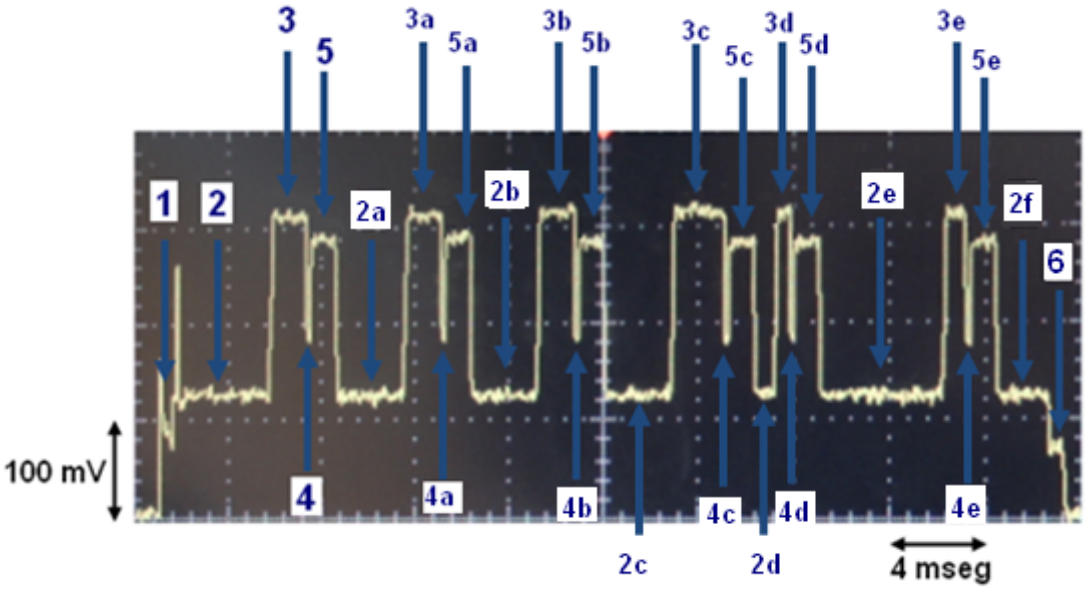

Imagem 9.3: Análise de consumo por ciclo de transmissão no osciloscópio digital

Identificação	***Descrição***	***Consumo***	***Duração***
1	*MCU em modo activo em execução a 16MHz*	*8 mA*	*0.64 ms*
2	*MCU a executar a 32 MHz*	*12.5 mA*	*4.16 ms*
3	*CSMA/CA. Transceptor em modo RX*	*31 mA*	*1.6 ms*
4	*Transição de RX para TX*	*18 mA*	*0.16 ms*
5	*Transmissão da mensagem. Modo TX*	*29 mA*	*1.12 ms*
6	*Fim da sequência (PM0 para PM2). MCU em execução a 16MHz*	*7 mA*	*0.64 ms*

Tabela 9.2: Descrição das fases do BlindNode

9.1.4. Cálculo do consumo e duração prevista

As mensagens enviadas pelo módulo móvel são constituídas por *payload APS* nulo sendo constituídas por apenas 33 *bytes*:

- 6 *bytes* de cabeçalho da camada física;
- 11 *bytes* de cabeçalho e *MFR* da camada *MAC*;
- 8 *bytes* de cabeçalho da camada *NWK*;
- 8 *bytes* de cabeçalho da camada de aplicação;

O valor esperado para a duração do envio de um *blast* seria cerca de 1,056*ms*, na prática verificou-se um valor de 1,12*ms* o que se encontra de acordo com os valores esperados.

$$Taxa\,de\,transmissão: 250\text{Kb}/s = 31{,}25\ Kbytes/s$$

$$Duração\_da\_mensagem = \frac{33}{31{,}25 Kbytes} = 1{,}056ms$$

O tempo de execução do intervalo (3) da Imagem 9.3, representa a execução do processo *CSMA/CA*. A duração deste processo pode variar conforme o ruído do canal. Isto deve-se a uma espera de um intervalo aleatória quando o canal é detectado como ocupado ou com muito ruído. Através de uma análise de 10 observações verificou-se uma variação no processo *CSMA/CA* entre 0,6*ms* e 2,40*ms,* com uma média de 1,34*ms*.

O consumo energético por cada ciclo é dado por:

$$Consumo\_por\_ciclo(mA \times ms) = \sum_{i=0}^{i=Ntotal} (I_i(mA) \times \Delta t_i(ms)) + I_{Sleep} \times SleepTime$$

Onde *i* corresponde a cada uma das sub-fases de funcionamento, *Ii* e *Δti* à corrente e intervalo de funcionamento dessas sub-fase. *Isleep* corresponde à corrente consumida em *SleepMode* (0,5*μA)* e *SleepTime* ao tempo que o módulo irá permanecer nesse estado.

O consumo diário é dado por:

$$Consumo\_diário(mAH) = Consumo\_por\_ciclo \times \frac{60000(ms)}{\Delta t_{ciclo}(ms)} \times 60(\text{min}) \times 24(horas) / \frac{1000(ms/s)}{3600(s/Hora)}$$

Os valores referidos nas tabelas seguintes são o resultado da média de 10 mensagens analisadas. Nas tabelas 9.3 e 9.4 são referidos os consumos energéticos do módulo móvel por ciclo de transmissão e consumo diário de acordo com diferentes intervalos de

SleepTime. Nas tabelas 9.5 e 9.6 são referidas as durações previstas dos módulos, em dias, de acordo com diferentes baterias.

	ConsumoPorCiclo (mA*ms)		
Sleep Cycle(ms)	*1 Blast*	*5 Blasts*	*8 Blasts*
3000	173,8	766,28	1144,7
5000	174,8	767,28	1145,7
10000	177,3	769,78	1148,2
20000	182,3	774,78	1153,2
60000	202,3	794,78	1173,2

Tabela 9.3: Consumo por ciclo de envio

	Consumo Diário (mAH)		
Sleep Cycle(ms)	*1Blast*	*5Blasts*	*8Blasts*
3000	1,39	6,05	8,98
5000	0,84	3,65	5,44
10000	0,43	1,84	2,74
20000	0,22	0,93	1,38
60000	0,08	0,32	0,47

Tabela 9.4: Consumo diário

	Duração (dias)		
	650mAH		
Sleep Cycle(ms)	*1 Blast*	*5 Blasts*	*8 Blasts*
3000	469,2	107,4	72,4
5000	776,4	177,8	119,6
10000	1529,2	353,2	237,3
20000	2972,9	700,5	471,1
60000	8034,1	2045,9	1386,4

Tabela 9.5: Duração de vida de um BlindNode com bateria de 650mAH

	Duração (dias)					
	2300mAH			*3100mAH*		
Sleep Cycle(ms)	*1 Blast*	*5 Blasts*	*8 Blasts*	*1 Blast*	*5 Blasts*	*8Blasts*
3000	1660,3	380,0	256,0	2237,8	512,1	345,1
5000	2747,3	629,3	423,1	3702,9	848,2	570,3
10000	5411,2	1249,7	839,5	7293,3	1684,4	1131,5
20000	10519,7	2478,6	1666,9	14178,7	3340,7	2246,7
60000	28428,4	7239,3	4905,9	38316,5	9757,3	6612,3

Tabela 9.6: Duração de vida de um BlindNode com bateria de 2300mAH e 3100mAH

9.1.5. Comparação com o sistema *ZigBee location engine*

O sistema implementado foi baseado no *ZigBee location engine* (3.3.7.). No sistema baseado no *location engine, os BlindNodes* enviam 8 *blasts*, seguidos de uma mensagem a informar o fim do ciclo de envio. Os *RefNodes* ao receberem esses *blasts*, processam-os e respondem com uma mensagem para o *BlindNode* com a posição e valores de *RSSI* registados. O *BlindNode* responde com *acknowledges*. No exemplo figurado em 9.4 apenas 3 *RefNodes* detectaram o *BlindNode*. Em seguida o *BlindNode* espera um determinado intervalo de tempo e envia uma mensagem para o *LocDongle* a informar a sua própria localização.

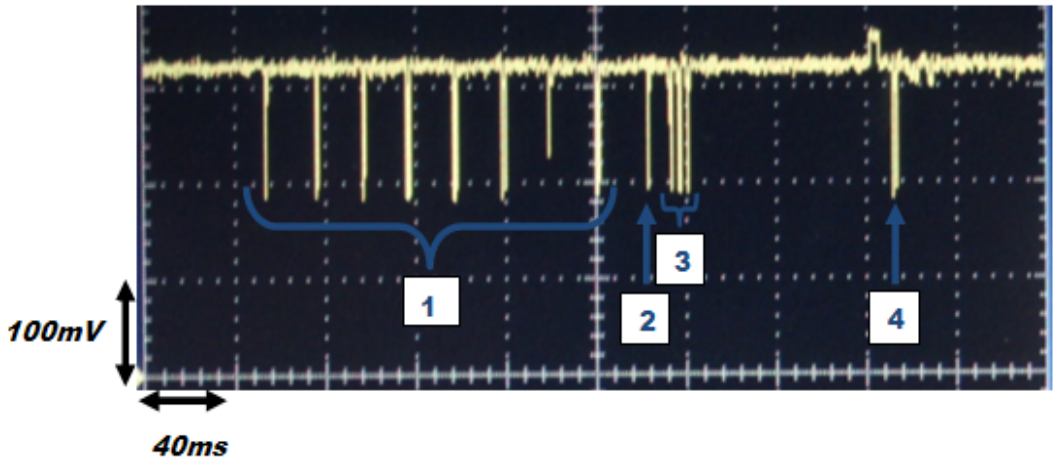

Imagem 9.4: Consumo energético do Location Engine

Descrição:

- 1- envio de 8 *blasts* (*BLIND_NODE_BLAST_DELAY = 20ms*);
- 2- envio de mensagem a informar o fim do envio de *blasts;*
- 3- *acknowledgs* transmitidos pelo *BlindNode para os 3 RefNodes*;
- 4 - envio da mensagem com a localização do *BlindNode para o LocDongle*.

9.1.6. Comparação energética com o sistema *Location Engine*

Nas imagens 9.5 e 9.6 são apresentados os gráficos dos consumos energéticos entre o sistema *Location Engine* e o sistema implementado neste projecto. É notória a diferença do consumo energético entre os dois sistemas. A grande diferença entre ambos diz respeito a uma escuta permanente por parte do *BlindNode* de mensagens de configuração. O sistema *location engine* tem imensas vantagens em relação ao sistema implementado entre elas,

configuração do ciclo de envio dos *BlindNodes* e configuração da posição dos *RefNodes on demand* do utilizador, mas por outro lado exige uma escuta desnecessária por parte dos *BlindNodes*. Mesmo que o *BlindNode* adormecesse durante o ciclo entre envio de *blasts* o sistema implementado revelaria mesmo assim notórias vantagens em relação ao consumo energético dos módulos. Isto devido à troca de mensagens necessário entre *BlindNode e RefNodes* o que exige um tempo muito superior em *PM0*, que por sua vez implica um maior consumo energético. Na tabela 9.7 é apresentado uma comparação quantitativa entre os consumos energéticos dos dois sistemas.

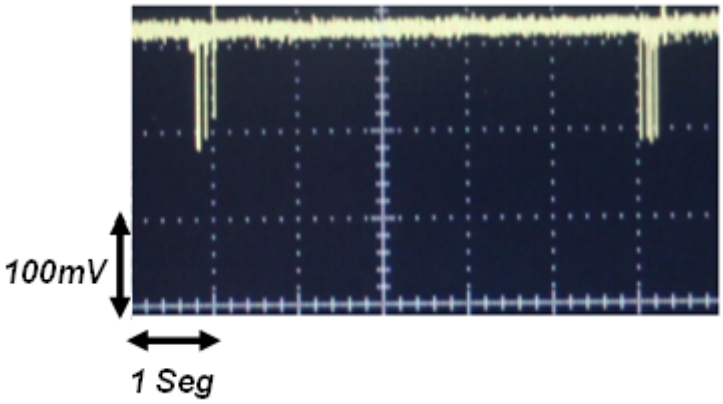

Imagem 9.5: Consumo energético do sistema Location Engine

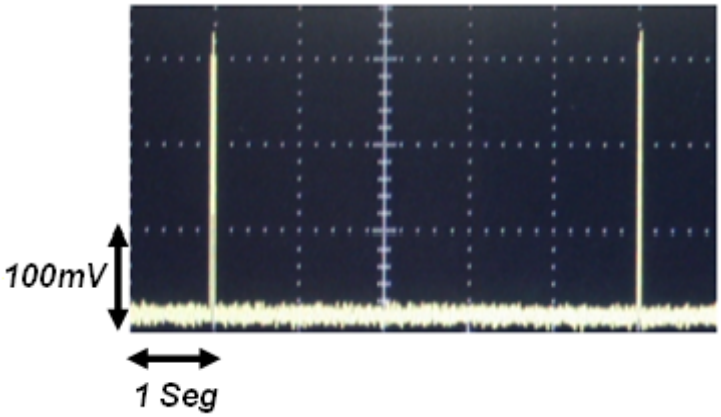

Imagem 9.6: Consumo energético do sistema implementado

Melhoria da duração de vida útil das baterias (nº de vezes) entre o sistema e o Location Engine			
Sleep Cycle (ms)	**8 blasts**	**5 blasts**	**1 blast**
3000	82,8	122,8	536,8
5000	136,8	203,5	888,4
10000	271,5	404,2	1750,1
20000	539,2	801,7	3402,6
60000	1586,9	2341,7	9195,7

Tabela 9.7: Comparação entre o consumo energético do sistema implementado e o ZigBee Location Engine

9.2 . Exactidão e precisão da localização

Os resultados apresentados são baseados em dois modelos de estudo tendo em consideração a alteração das antenas dos quatro módulos de referência. O primeiro consiste no uso de quatro antenas *Titanis 2,4GHz* (dipolo de meia onda) com orientação horizontal e o segundo em quatro antenas impressas desenvolvidas. Para cada modelo analisado foi realizado um estudo de localização por proximidade e outro através de uma rede neuronal.

Para o processo de localização por proximidade foi considerada uma variação da orientação e da altura do *BlindNode*. Deste modo foi considerado uma localização com a antena do *BlindNode* (*Titanis 2,4GHz)* na vertical e na horizontal para 1,5m e 2m de altura. A média destas quatro considerações permitiu-nos estimar a percentagem em que o *BlindNode* foi localizado pelo *RefNode* mais próximo. O resultado obtido foi baseado numa média de 10 medições para cada um dos casos referidos.

A localização através da rede neuronal foi obtida através de uma calibração de 132 pontos, sendo estes resultantes de uma média de 10 *fingerprints* para cada ponto. Neste processo foi considerada a orientação vertical do *BlindNode* e uma média de 10 valores para cada ponto de calibração.

Com o uso das antenas *Titanis* 2,4*GHz* do *kit* de desenvolvimento nos *RefNodes* foi obtido uma localização correcta pelo *RefNode* mais próximo em 65% das medições; com o uso da rede neuronal foi obtido uma exactidão inferior a 3,5 metros em 75% das medições, e uma exactidão inferior a 2,5m para 60% das mesmas. Com o uso de 8 antenas de referência no mesmo espaço de localização é mencionada uma resolução inferior a 2

metros, referido em [8].

Com o uso das antenas impressas desenvolvidas foi obtido uma localização correcta pelo *RefNode* mais próximo em cerca de 85% das medições; com o uso da rede neuronal foi obtido uma exactidão inferior a 3,5 metros em 78% das medições, e uma exactidão inferior a 2,5m para 64% das mesmas.

O processo de localização apresentou vários factores de variação, entre eles: altura do *BlindNode*, orientação das antenas, posição do *BlindNode* no cliente, local no ambiente de localização, presença de muitas pessoas.

Capítulo 10

Conclusões e trabalho futuro

Como se pode verificar pela análise do capítulo 9 a duração de vida útil dos módulos móveis com a programação implementada neste sistema pode variar desde vários meses a vários anos, conforme a aplicação e necessidade do sistema implementado. Para o sistema implementado foi considerado um envio cíclico de 5 *blasts* em cada 5 segundos utilizando uma bateria de 650mAH. A duração prevista é cerca de 6 meses. Já foi iniciado o teste para comprovar a veracidade destes valores mas ainda continua a transmitir sem problemas.

Em relação ao processo de localização foram elaboradas duas soluções: uma sem necessidade de calibração baseada em proximidade e outra de maior exactidão baseada em redes neuronais mas necessitando de uma calibração prévia. É de referir que o protocolo de comunicação desenvolvido proporciona a base para uma localização baseada em *fingerprinting* assim como para triangulação, sendo a rede neuronal apenas um solução possível. A implementação das antenas impressas desenvolvidas permitiu melhorar significativamente o desempenho do processo de localização por proximidade

obtendo-se uma localização correcta em cerca de 85% das medições com o uso das antenas desenvolvidas comparativamente a uma localização correcta em 65% das medições com as antenas *Titanis 2,4GHz*. Com o uso da rede neuronal foi verificado uma ligeira melhoria no processo de localização com o uso das antenas desenvolvidas (antenas impressas) em relação às antenas *Titanis 2,4GHz*.

No âmbito de sistemas de localização de baixo consumo existe imenso trabalho futuro que se poderá realizar, tanto a nível de exactidão da localização como a nível de melhoria do consumo energético. O trabalho futuro no âmbito deste dissertação poderia ser direccionado para:

- Configuração do *sleep time* no início do registo na rede, ou através de um modo habilitado de *beacons*. No segundo caso a programação é possível mas ter-se-à que ter em atenção o facto do processo de escuta inerente ao modo *beacon* habilitado consumir mais do que o processo de transmissão;
- Estudo aprofundado sobre as melhores antenas para o sistema de localização;
- Estudo da disposição e orientação das antenas para ambientes *indoor* de modo a obter a melhor exactidão e precisão do sistema;
- Associação de novos parâmetros ao processo de localização como *AoA*;
- Uso de tecnologia *SoC* de menor consumo energético (exemplo, *CC1111*);
- Alteração da programação da comunicação da rede de sensores do sistema de modo a possibilitar uma actualização mais rápida da localização;
- Melhoria do ambiente gráfico e suas funcionalidades;
- Alteração do algoritmo de localização, em [24] indicam que algoritmos probabilísticos apresentaram melhores valores que algoritmos determinísticos (*sugestão: Rede de Bayes*);
- Associação de interrupções ao módulo móvel que possibilitem um adormecimento em *Deep Sleep* em vez de *Lite Sleep*, conservando assim a duração de vida da bateria;
- Testar a localização baseada em *RSSI* associada a uma correlação média entre os símbolos recebidos e não apenas em *RSSI*.

Anexos

Programação do BlindNode:

CONSTANTES:

```
// TODAS AS VARIAVEIS ENCONTRAM-SE EM MILISSEGUNDOS
// tempo em que o Blind permanecerá em Sleep Mode
#define SLEEP_TIME                          5000
// delay entre envio de Blasts
#define BLINDNODE_BLAST_DELAY          0
// número de Blasts enviados por ciclo
#define BLINDNODE_BLAST_COUNT          5

/*********************************************************************
 * @fn      BlindNode_ProcessEvent
 * @brief   Aplicação genérica que processa a sequência agendada de eventos (task  events)
 * @param   task_id  - OSAL  task ID
 * @param   events – eventos para processar
 * @return  none
 */
uint16 BlindNode_ProcessEvent( uint8 task_id, uint16 events )
{
 if ( events & BLINDNODE_BLAST_EVT ) // fase 2
 {
  if ( blastCnt == 0 )  //fase2.3
   SleepModeFunction();
  else
  {
   afAddrType_t dstAddr;
   uint8 stat, delay;
   // envio de BroadCast para todos os RefNodes
   dstAddr.addrMode = afAddrBroadcast;
   dstAddr.addr.shortAddr = NWK_BROADCAST_SHORTADDR_DEVALL;
   dstAddr.endPoint = LOCATION_REFNODE_ENDPOINT;
   if ( --blastCnt == 0 ) //fase 2.2
   {
    // envio da mensagem indicadora de fim de envio de blasts
     stat = AF_DataRequest( &dstAddr, (endPointDesc_t *)&epDesc,
                        END_OF_BLASTS_IDENTIFICATION, 0, NULL,&transId, AF_SKIP_ROUTING, 1 );
     delay = BLINDNODE_BLAST_DELAY;
   }
   else //fase 2.1
   {
    // envio de uma mensagem blast
     stat = AF_DataRequest( &dstAddr, (endPointDesc_t *)&epDesc,
                        LOCATION_RSSI_BLAST, 0, NULL,&transId, AF_SKIP_ROUTING, 1 );
    delay = BLINDNODE_BLAST_DELAY;
   }
   // incremento de blastCnt caso se verifique falha no envio da mensagem
   if ( stat != afStatus_SUCCESS )
   {
    blastCnt++;
   }
   // execução do evento BLINDNODE_BLAST_EVENT  após "delay" milissegundos
   osal_start_timerEx( BlindNode_TaskID, BLINDNODE_BLAST_EVT, delay );
 }
 return ( events ^ BLINDNODE_BLAST_EVT );
}
// evento para inicio da sequência de envio de Blasts
if ( events & BLINDNODE_START_EVT )  // fase1
{
```

```
    startBlast();
    return ( events ^ BLINDNODE_START_EVT );
   }
   return 0;  // Descarta eventos não conhecidos
  }


  /*****************************************************************
   * @fn      SleepModeFunction
   * @brief   função que permite a entrada em sleep mode
   * @param   none
   * @return  none
   */

  static void SleepModeFunction( void )  // fase 2.3.1
  {
     // Desliga o receptor RF
   if ((ZDO_Config_Node_Descriptor.CapabilityFlags & CAPINFO_RCVR_ON_IDLE) == 0)
   {
    uint8 x;
    x = false;
    ZMacSetReq( ZMacRxOnIdle, &x );
   }
  // Activação do modo sleep
   SampleApp_Sleep( TRUE );

  // processamento de BLINDNODE_START_EVT após SLEEP_TIME  milissegundos
   osal_start_timerEx( BlindNode_TaskID, BLINDNODE_START_EVT, SLEEP_TIME );
  }


  /*****************************************************************
   * @fn      startBlast
   * @brief   função que inicia a sequência do envio de blasts
   * @param   none
   * @return  none
   */
  static void startBlast( void )   // fase 1.1
  {
  // parâmetros de configuração
   afAddrType_t dstAddr;
   dstAddr.addrMode = afAddrBroadcast;
   dstAddr.addr.shortAddr = NWK_BROADCAST_SHORTADDR_DEVALL;
   dstAddr.endPoint = LOCATION_REFNODE_ENDPOINT;

  // reactivação do modo activo com activação da Led
   SampleApp_Sleep( FALSE );
   HalLedSet(HAL_LED_1,HAL_LED_MODE_ON);

  // envio de uma Blast message
   (void)AF_DataRequest( &dstAddr, (endPointDesc_t *)&epDesc,
                    LOCATION_RSSI_BLAST, 0,NULL, &transId, AF_SKIP_ROUTING, 1 );

  // processamento de BLINDNODE_BLAST_EVT  após BLINDNODE_BLAST_DELAY  milissegundos
   blastCnt = BLINDNODE_BLAST_COUNT;
   osal_start_timerEx( BlindNode_TaskID, BLINDNODE_BLAST_EVT, BLINDNODE_BLAST_DELAY );
  }
```

Programação do RefNode:

CONSTANTES:

```
// Array default de roteamento de mensagens do sistema
static uint16 ARRAY_REFNODES[] = {0x0001,0x143E,0x287B,0x3CB8,0x0000,0x0000,0x0000,0x0000,0x0000};
// Offset de RSSI
static byte OFFSET = 50;

/*********************************************************************
 * @fn      BlindNode_ProcessEvent
 * @brief   Aplicação genérica que processa a sequência agendada de eventos (task  events)
 * @param   task_id  - OSAL  task ID
 * @param   events – eventos para processar
 * @return  none
 */
uint16 RefNode_ProcessEvent( byte task_id, uint16 events )
{
 if ( events & SYS_EVENT_MSG )
 {
  afIncomingMSGPacket_t *MSGpkt;

  // os eventos serão processados consoante o tipo de mensagem recebida
  while ((MSGpkt = (afIncomingMSGPacket_t *)osal_msg_receive(RefNode_TaskID)))
  {
   switch ( MSGpkt->hdr.event )
   {
  // evento despoletado quando é realizado o envio de uma mensagem
   case AF_DATA_CONFIRM_CMD: //fase 2
    {
     afDataConfirm_t *afDataConfirm = (afDataConfirm_t *)MSGpkt;
     // se não receber acknowledge envia a informação do endereço desse RefNode para o Coordinator
     // mensagem  TOKEN_NOT_RECEIVED
     if ( afDataConfirm->hdr.status != ZSuccess )
     {
      byte NotDetectedRefNodes[2];
      if ( afDataConfirm->hdr.status == ZMacNoACK )
      {
       afAddrType_t dstAddr;
       dstAddr.addrMode = afAddr16Bit;
       dstAddr.addr.shortAddr = 0x0000;
       dstAddr.endPoint = LOCATION_DONGLE_ENDPOINT;

       NotDetectedRefNodes[0] = LO_UINT16(ARRAY_REFNODES[INDICE+1]);
       NotDetectedRefNodes[1] = HI_UINT16(ARRAY_REFNODES[INDICE+1]);

       (void)AF_DataRequest( &dstAddr, (endPointDesc_t *)&epDesc,
                   TOKEN_NOT_RECEIVED, ADDRESS_SIZE, NotDetectedRefNodes,
                   &transId, AF_SKIP_ROUTING, 3);
      }
     }
    }
    break;
    // evento despoletado quando é recebido uma mensagem de dados
   case AF_INCOMING_MSG_CMD: // fase 1
    processMSGCmd( MSGpkt );
    break;
   osal_msg_deallocate( (uint8 *)MSGpkt );
  }
  // Return unprocessed events.
  return ( events ^ SYS_EVENT_MSG );
 }
 return 0;  // descarta eventos desconhecidos
}
```

```
/*********************************************************************
* @fn      processMSGCmd
* @brief   Processa as mensagens de dados
* @param   none
* @return  none
*/
static void processMSGCmd( afIncomingMSGPacket_t *pkt ) // fase 1
{
// consoante o ClusterID da mensagem irá ser despoletado uma função diferente
 switch ( pkt->clusterId )
 {
  // recepção da mensagem após envio de sequência de blasts,  END_OF_BLASTS_IDENTIFICATION
 case END_OF_BLASTS_IDENTIFICATION: // fase 1.2
  LocalListActualization( pkt );
  break;

  //  recepção de uma mensagem blast,  LOCATION_RSSI_BLAST
 case LOCATION_RSSI_BLAST: // fase 1.1
  addBlast( pkt->srcAddr.addr.shortAddr, pkt->LinkQuality );
  break;

  // recepção de um mensagem HIRSSIFLOWTABLEMESSAGE
 case HIRSSIFLOWTABLEMESSAGE:  // fase 1.2.1
  FlowListActualizationByHiRSSI(pkt->cmd.Data,pkt->cmd.DataLength);
  break;

  // recepção de um mensagem NEURONALFLOWTABLEMESSAGE
 case NEURONALFLOWTABLEMESSAGE: // fase 1.2.2
  FlowListWithAllRssiValues(pkt->cmd.Data, pkt->cmd.DataLength);
  break;

  // recepção de uma mensagem ROUTECONFIGURATIONS
 case ROUTECONFIGURATIONS: // fase 1.3
  CreateNewRoute(pkt->cmd.Data,pkt->cmd.DataLength);
  break;

 default:
  break;
 }
}


/*********************************************************************
* @fn      addBlast
*  @brief   Actualiza ou cria uma estrutura numa lista ligada com endereço e
* RSSI recebidos para um determinado Blind Addr
* @param   uint16 – Network Addr do Blind Node
* @param   byte   - RSSI da mensagem recebida do Blind Node
* @return  none
*/

static void addBlast( uint16 addr, byte RSSI )
{
 Local_Blind_Element *ptr = blastPtr;
 // procura de "matching" do blast recebido com os endereço dos Blinds da lista
 while ( ptr )
 {
  if ( ptr->LocalBlindAddr == addr )
  }
    break;
  }
  ptr = ptr->next;
 }
 RSSI = (byte)(RSSI+OFFSET);
```

```
  // se houver matching na lista adiciona RSSI e incrementa contador
  if ( ptr )
  {
    ptr->SumOfRSSI += RSSI;
    ptr->cnt++;
  }
  // alocação de memória para um novo Blind na lista
  else
  {
    ptr = (Local_Blind_Element *)osal_mem_alloc( sizeof( Local_Blind_Element ) );

    if ( ptr )
    {
      ptr->next = blastPtr;
      blastPtr = ptr;

      ptr->LocalBlindAddr = addr;
      ptr->SumOfRSSI = RSSI;
      ptr->cnt = 1;
    }
  }
}

/*********************************************************************
* @fn      LocalListActualization
* @brief   calcula a média de RSSI do Blind de endereço da mensagem recebida;
*  adiciona ou actualiza uma entrada do Blind à lista local BlindLocalFlowElements e
*  liberta a estrada do Blind da lista Local_Blind_Element
* @param   uint16 – mensagem recebida pelo RefNode
* @return  none
*/

static void LocalListActualization( afIncomingMSGPacket_t *pkt )
{
  BlindLocalFlowElements *Bptr = BlindFlowPtr;
  Local_Blind_Element *ptr = blastPtr;
  Local_Blind_Element *prev = NULL;

  // procura de "matching" do Blind recebido com os previamente contidos na lista  Local_Blind_Element
  while ( ptr )
  {
    if ( ptr->LocalBlindAddr == pkt->srcAddr.addr.shortAddr )
      break;

    prev = ptr;
    ptr = ptr->next;
  }
  if ( ptr )
  {
    // Actualização ou inicialização da Lista BlindLocalFlowElements com os dados do Blind
    while (Bptr)
    {
      if ( Bptr->BlindAddress == ptr->LocalBlindAddr )
        break;
      Bptr = Bptr->next;
    }
    // Actualização do valor de RSSI
    if (Bptr)
      Bptr->HiRSSI = (byte)((ptr->SumOfRSSI)/(ptr->cnt));
    else
    {
      // Inicialização de uma nova entrada na Lista  BlindLocalFlowElements
      Bptr = (BlindLocalFlowElements *)osal_mem_alloc( sizeof(BlindLocalFlowElements) );
      if ( Bptr )
```

```
        {
          // Actualização da lista  BlindLocalFlowElements
          Bptr->next = BlindFlowPtr;
          BlindFlowPtr = Bptr;
          Bptr->BlindAddress = ptr->LocalBlindAddr;
          Bptr->HiRSSI = (byte)((ptr->SumOfRSSI)/(ptr->cnt));
          //Bptr->RefNodeAddress = _NIB.nwkDevAddress;
        }
       }

       // remoção do elemento da lista ligada Local_Blind_Element
       if ( prev )
        prev->next = ptr->next;
       else
        blastPtr = ptr->next;

       osal_mem_free( ptr );
      }
     }


     /*********************************************************************
     * @fn      FlowListActualizationByHiRSSI
     * @brief   envia a mensagem  HIRSSIFLOWTABLEMESSAGE actualizada para o próximo nó com a indicação de
     * BlindNode addr, RefNode de maior detecção de RSSI e valor de RSSI correspondente
     * @param  * byte – mensagem recebida
     * @param   uint16 – tamanho da mensagem recebida
     * @return  none
     */
     static void FlowListActualizationByHiRSSI( byte *msg, uint16 msgLength )
     {
      afAddrType_t dstAddrTeste;
      BlindLocalFlowElements *ReceivedPtr = ReceivedBlindFlowPtr;

      // numero de Blinds detectado até ao nó anterior
      int ReceivedBlinds;
      ReceivedBlinds = (msgLength / LIST_ELEMENT_SIZE);

      // passagem da mensagem recebida do nó anterior para uma lista ligada  BlindLocalFlowElements
      for (int i=0; i < ReceivedBlinds; i++)
      {
       ReceivedPtr = (BlindLocalFlowElements *)osal_mem_alloc( sizeof(BlindLocalFlowElements) );
       if ( ReceivedPtr )
       {
        ReceivedPtr->next = ReceivedBlindFlowPtr;
        ReceivedBlindFlowPtr = ReceivedPtr;
        ReceivedPtr->BlindAddress = BUILD_UINT16( msg[i*LIST_ELEMENT_SIZE + 1],
     msg[i*LIST_ELEMENT_SIZE + 2] );
        ReceivedPtr->RefNodeAddress = BUILD_UINT16( msg[i*LIST_ELEMENT_SIZE + 3],
     msg[i*LIST_ELEMENT_SIZE + 4] );
        ReceivedPtr->HiRSSI = msg[i*LIST_ELEMENT_SIZE + 5];
       }
      }

      BlindLocalFlowElements *Bptr = BlindFlowPtr;

      // actualização ou adição dos Blinds da lista local Local_Blind_Element na lista de envio BlindLocalFlowElements
      while(Bptr)
      {
       bool addBlind = true;
       BlindLocalFlowElements *ReceivedPtrAux = ReceivedBlindFlowPtr;

       BlindLocalFlowElements *RemovePtr = NULL;
```

```
 // actualização da lista
 while( ReceivedPtrAux )
 {
  if (Bptr->BlindAddress == ReceivedPtrAux->BlindAddress)
  {
    addBlind = false;
    if (Bptr->HiRSSI > ReceivedPtrAux->HiRSSI)
    {
      ReceivedPtrAux->RefNodeAddress = _NIB.nwkDevAddress;
      ReceivedPtrAux->HiRSSI = Bptr->HiRSSI;
    }
    break;
  }
  ReceivedPtrAux = ReceivedPtrAux->next;
 }

 // adição a lista ligada
 if (addBlind)
 {
  ReceivedPtrAux = (BlindLocalFlowElements *)osal_mem_alloc( sizeof(BlindLocalFlowElements) );
  ReceivedPtrAux->next = ReceivedBlindFlowPtr;
  ReceivedBlindFlowPtr = ReceivedPtrAux;
  ReceivedPtrAux->BlindAddress = Bptr->BlindAddress;
  ReceivedPtrAux->HiRSSI = Bptr->HiRSSI;
  ReceivedPtrAux->RefNodeAddress = _NIB.nwkDevAddress;
 }
 // libertação de memória
 RemovePtr = Bptr;
 Bptr = Bptr->next;
 osal_mem_free( RemovePtr );
}

BlindFlowPtr = NULL;

BlindLocalFlowElements *SendingPtr = ReceivedBlindFlowPtr;
int blinds_flowing=0;

// conversao da lista para array de envio
while (SendingPtr)
{
 BlindLocalFlowElements *RemovePtrAux = NULL;
 SendingFlowList[blinds_flowing*LIST_ELEMENT_SIZE +1] = LO_UINT16(SendingPtr->BlindAddress);
 SendingFlowList[blinds_flowing*LIST_ELEMENT_SIZE +2] = HI_UINT16(SendingPtr->BlindAddress);
 SendingFlowList[blinds_flowing*LIST_ELEMENT_SIZE +3] = LO_UINT16(SendingPtr->RefNodeAddress);
 SendingFlowList[blinds_flowing*LIST_ELEMENT_SIZE +4] = HI_UINT16(SendingPtr->RefNodeAddress);
 SendingFlowList[blinds_flowing*LIST_ELEMENT_SIZE +5] = SendingPtr->HiRSSI;
 blinds_flowing++;

 RemovePtrAux = SendingPtr;
 SendingPtr = SendingPtr->next;
 // libertação de memória
 osal_mem_free( RemovePtrAux );
}

SendingFlowList[0]=blinds_flowing;
ReceivedBlindFlowPtr = NULL;

// envio da mensagem para o próximo nó (RefNode ou LocDongle)
if (ARRAY_REFNODES[INDICE+1] != 0x0000)
{
 dstAddrTeste.addrMode = afAddr16Bit;
 dstAddrTeste.addr.shortAddr = ARRAY_REFNODES[INDICE+1];
 dstAddrTeste.endPoint = LOCATION_REFNODE_ENDPOINT;
}
```

```
  else
  {
    dstAddrTeste.addrMode = afAddr16Bit;
    dstAddrTeste.addr.shortAddr = 0x0000;
    dstAddrTeste.endPoint = LOCATION_DONGLE_ENDPOINT;
  }

  (void)AF_DataRequest( &dstAddrTeste, (endPointDesc_t *)&epDesc, HIRSSIFLOWTABLEMESSAGE,
blinds_flowing*LIST_ELEMENT_SIZE+1, SendingFlowList, &transId, AF_SKIP_ROUTING, 1 );
}



/*********************************************************************
* @fn      FlowListWithAllRssiValues
* @brief   envia a mensagem  NEURONALFLOWTABLEMESSAGE actualizada para o próximo nó com a
* indicação de BlindNode addr, RefNodes que o detectaram o Blind e seus valores de RSSI médios
* @param  * byte – mensagem recebida
* @param   uint16 – tamanho da mensagem recebida
* @return  none
*/

static void FlowListWithAllRssiValues( byte *msg,uint16 msgLength )
{
  afAddrType_t dstAddrTeste;

  // Endereço do Blind para detecção
  uint16 BlindAddressOfNeuronal = BUILD_UINT16( msg[0],msg[1] );
  // Numero de Refnodes referidos na mensagem recebida
  int NumberOfPreviousRefNodes = msg[2];

  // Cria um novo array clone da mensagem recebida
  for (int i=0; i<msgLength;i++)
    MessageWithAllRSSI[i] = msg[i];

  BlindLocalFlowElements *Bptr = BlindFlowPtr;
  BlindLocalFlowElements *RemovePtr = NULL;

// procura por um matching dos elementos da lista local com endereço do Blind que se pretende localizar
  while(Bptr)
  {
    if (BlindAddressOfNeuronal == Bptr->BlindAddress)
    {
 MessageWithAllRSSI[BLINDFIELDOFFSET + NumberOfPreviousRefNodes*REFNODEFIELDOFFSET]
 = LO_UINT16(_NIB.nwkDevAddress);
  MessageWithAllRSSI[BLINDFIELDOFFSET + NumberOfPreviousRefNodes*REFNODEFIELDOFFSET + 1]
 = HI_UINT16(_NIB.nwkDevAddress);
     MessageWithAllRSSI[BLINDFIELDOFFSET + NumberOfPreviousRefNodes*REFNODEFIELDOFFSET + 2]
= Bptr->HiRSSI;
      NumberOfPreviousRefNodes++;
      MessageWithAllRSSI[2] = NumberOfPreviousRefNodes;
      break;
    }

// actualização e remoção do elemento da lista
    RemovePtr = Bptr;
    Bptr = Bptr->next;
    osal_mem_free( RemovePtr );
  }

  BlindFlowPtr = NULL;

  // envio da mensagem para o próximo nó (RefNode ou LocDongle)
  if (ARRAY_REFNODES[INDICE+1] != 0x0000)
```

```
      {
        dstAddrTeste.addrMode = afAddr16Bit;
        dstAddrTeste.addr.shortAddr = ARRAY_REFNODES[INDICE+1];
        dstAddrTeste.endPoint = LOCATION_REFNODE_ENDPOINT;
      }
      // envio da ultima mensagem para o Dongle
      else
      {
        dstAddrTeste.addrMode = afAddr16Bit;
        dstAddrTeste.addr.shortAddr = 0x0000;
        dstAddrTeste.endPoint = LOCATION_DONGLE_ENDPOINT;
      }

    (void)AF_DataRequest(&dstAddrTeste,(endPointDesc_t*)&epDesc,NEURONALFLOWTABLEMESSAGE,
BLINDFIELDOFFSET + NumberOfPreviousRefNodes*REFNODEFIELDOFFSET, MessageWithAllRSSI,
    &transId, AF_SKIP_ROUTING, 1 );
    }


    /*********************************************************************
    * @fn      CreateNewRoute
    * @brief   configura a nova rota de circulação dos RefNodes
    * @param  * byte – mensagem recebida
    * @param   uint16 – tamanho da mensagem recebida
    * @return  none
    */
    static void CreateNewRoute( byte *msg, uint16 msgLength ){
      int REFNODES_IN_FLOWLIST = msgLength/2;
      for (int i = 0; i < REFNODES_IN_FLOWLIST; i++)
        ARRAY_REFNODES[i] = BUILD_UINT16( msg[i*ADDRESS_SIZE],msg[i*ADDRESS_SIZE + 1] );

      ARRAY_REFNODES[REFNODES_IN_FLOWLIST] = 0x0000;
      IndiceCalculation();
    }

    /*********************************************************************
    * @fn      IndiceCalculation
    * @brief   calcula o índice do RefNode da lista
    * @return  none
    */
    static void IndiceCalculation(void)
    {
      for (int i = 0; i < MAX_REFNODES; i++)
        if (ARRAY_REFNODES[i] == _NIB.nwkDevAddress)
          INDICE = i;
    }
```

Referência bibliográfica

[1] **P. Moore and P.Crossley**, *"GPS applications in power systems: Introduction to gps"*, Power Engineering Journal 13, no. 1, 33-39 (1999)

[2] **A. Leick**, *"Glonass satellite surveyving"*, Journal of Surveying Engineering 124 , nº 2, 91-99 (1998)

[3] **V. Zeimpekis, G.M. Giaglis, and G. Lekakos**, *"A taxonomy of indoor and outdoor positioning techniques for mobile location services"*, Tech. report, Athens University of Economics and Business, Department of Management Science & Technology, 2002

[4] **R.J Orr and G.D. Abowd**, *"The smart floor: A mechanism for natural user indetification and tracking"*, Proceedings of Human Factors in Computing Systems (Hague Netherlands) 275-276 Abril 2000

[5] **ZigBee Standards Organization**, *"ZigBee Specifications"* Dezembro 2006

[6] **K. Aamodt**, *"CC2431 Location Engine, Application Note AN042, Chipcon Products from Texas Instruments"*

[7] **Hui Liu, Darabi, H., Banerjee, P., Jing Liu,** *"Survey of Indoor Positioning Techniques and Systems"*, Part C: Applications and Reviews, IEEE Transactions on Volume 37, Issue 6, Page(s):1067 – 1080 Novembro 2007

[8] **Arnaldo Monteiro**, *"Projecto LOPES - máquina de localização"*, Instituto de Telecomunicações de Aveiro 2008

[9] **Hermevaldo Pereira Reis, Francisco Javier Ramirez Fernandez,** *"Análise das perdas de pacotes alterando as variáveis de distância, obstáculos e interferência electromagnética utilizando IEEE 802.15.4"*, Universidade de São Paulo, 2007

[10] **S. Pandey and P. Agrawal,** *"A survey on localization techniques for wireless networks"*, Journal of the Chinese Institute of Engineers 29 1125-1148 volume 29 nº7 2006

[11] **N.B Priyantha** *"The cricket indoor location system"*, Ph.D thesis, Massachusetts Institute of Technology Massachusetts Institute of Technology Junho 2005

[12] **C.K. Toh** *"Ah hoc mobile wireless networks:Protocols and systems"*,Prentice Hall 2002

[13] **Fernando Lobo Pereira** *"Sistemas e Veículos Autónomos - Aplicações na Defesa"*, Instituto de Defesa Nacional Setembro 2005

[14] **J. Krumm,S. Harris, B. Meyers, B. Brumitt, M. Hale, and S. Shafer "**Multi-Camera Multi-Person Tracking for EasyLiving" Third Workshop on Visual Surveillance

Redmond, WA, USA Julho 2000

[15] *A true System-on-Chip solution for 2,4GHZ IEEE 802.15.4/ZigBee, CC2430 Data Sheet* (rev 2.1) SWRS036F ChipconProducts from Texas Instruments

[16] Wikipédia, *http://pt.wikipedia.org/* visualizado em Março 2009

[17] **R. Want, A Hopper, V. Falção and J. Gibbons** *"The active badge location system"* Cambridge England 1992

[18] **P. Bahl and VN Padmanabhan** *"Radar: an in-building rf-based user location and tracking system"* Tel Aviv, Israel 2000

[19] **L.M. Ni, Y. Liu, Y.C. Lau, and A.P Patil** *"Landmarc: Indoor location sensing using active RFID Texas"*, USA Março 2003

[20] **M. Youssef and A. Agrawala** *"The horus location determination system"*, Wireless Networks 14 nº.3 357-374 2008

[21] **A. Ward, A. Jones, and A. Hopper** *"A new location technique for the active office"*, Personal Communications 1997

[22] **Dr. Rodney B. Waterhouse** *"Microstrip Patch Antennas A designer's Guide" Chapter 1, pag 7 a 10 , 2003*

[23] **Frank Winkler, Erik Fischer, Eckhard Grab, Gunter Fischer** *"A 60 GHz OFDM Indoor Localization System Based on DTDOA"* Humboldt University Berlin, Germany 2006

[24] **A.S. Krishnakumar, P. Krishnan** *"The theory and practice of Signal Strength-Based Location Estimation",* Avaya Labs Basking Ridge, New Jersey, USA 2005

[25] **NJhank,and DI Laurenson** *"Performance of a tdoa-aoa hybrid mobile location system"* London, United Kingdom Março 2001

[26] **S. Venkatraman and J. Caffery Jr** *"Hybrid toa/aoa techniques for mobile location in non-line-of-sight environments"* Proceedings of WCNC (Atlanta, USA) 2004

[27] *CC2431 DK Development Kit User Manual Rev. 1.5 Texas Instruments SWRU076D*

[28] **Shashank Tadakamadla** *"Indoor Local Positioning System for ZigBee, based on RSSI"* Mid Sweden University Outubro 2006

[29] **Rong Peng and Mihail L. Sichitiu** *"Angle of Arrival Localization for wireless sensor networks"* North Carolina State University 2006

[30] **Cassia Yuri Tatibana, Deisi Yuki Kaetsu** *"Uma introdução às Redes Neuronais" http://www.din.uem.br/ia/neurais* 2003

[31] **R. Battiti,M. Brunato, and A. Villani** *"Statistical learning theory for location*

fingerprinting in wireless LAN's", Outubro 2002

[32] **D. Madigan, E. Elnahrawy, R. Martin, W. Ju, P. Krishnan, and A.S. Krishnakumar** "*Bayesian Indoor Positionnig Systems*", Miami Florida USA, 2005

[33] **Constantine A. Balanis** *Antenna Theory Analysis and Design* 2ª Edition, New York pag 722-752 1997

[34] *http://boole.computer.org/,* visualizado em Março 2009

[35] *http://www.cl.cam.ac.uk/research/dtg/attarchive,* visualizado em Abril 2009

[36] **Lionel M. Ni, Yunhao Liu, Yiu Cho Lau, e Abhishek P. Patil** *"LANDMARC: Indoor Location Sensing Using Active RFID"* Hong Kong University of Science and Technologiy, China, pp 407-415 2003

[37] Ubisense, *http://www.ubisense.net* visualizado em Abril 2009

[38] Location Detection Solution for ZigBee Wireless Sensor Network – CC2431 and Z-Stack – Texas Instruments *http://embedded-system.net,* visualizado em 13 Junho 2007

[39] Location ZigBee nodes using TI's cc2431 location engine and the Daintree Networks Sensor Network Analyser Aplcation Note AN016

[40] TI, *http://www.ti.com/corp/docs/landing/cc2431/,* visualizado em Abril 2009

[41] ZigBee Aliance, *http://www..ZigBee.org/,* visualizado em Abril 2009

[42] **Joe Dvorak** "*IEEE 802.15.4 overview*", Motorola Setembro 2005

[43] *Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks* (LR-WPANS) IEEE 2006

[44] FANN, *http://leenissen.dk/fann,* visualizado em Abril de 2009

[45] TinyOS, *http://www.tinyos.net,* visualizado em Abril 2009

[46] TI, *http://focus.ti.com,* visualizado em Abril 2009

[47] PCB de múltiplas camadas operando em altas frequências, *www2.dbd.puc-rio.br/pergamum*, visualizado em Junho 2009

[48] Microstrip Patch Antenna Design and Results, *etd.lib.fsu.edu/theses/,* visualizado em Junho 2009

[49] Titanis 2,4GHz Swivel SMA Antenna, *www.semiconductorstore.com,* visualizado em Junho 2009