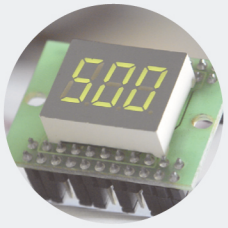
INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES






Monitorização e Registo Wireless de Medicação Intravenosa

FÁBIO ANDRÉ FERREIRA DA SILVA BORGES
julho de 2017

POLITÉCNICO
DO PORTO

Monitorização e Registo Wireless de Medicação Intravenosa

Fábio André Ferreira da Silva Borges
1110500

Tese/Dissertação

Departamento de Engenharia Electrotécnica

Instituto Superior de Engenharia do Porto

2017

Este relatório satisfaz, parcialmente, os requisitos que constam da Ficha de Disciplina de Tese/Dissertação , do 2º ano, do Mestrado em Engenharia Electrotécnica e de Computadores - Automação e Sistemas

Candidato: Fábio André Ferreira da Silva Borges,
Nº 1110500, 1110500@isep.ipp.pt

Orientação científica: Doutor, José Ricardo Teixeira Puga, jpt@isep.ipp.pt

Co-orientação científica: Doutor, Maria Judite Madureira da Silva Ferreira, mju@isep.ipp.pt

Empresa: ISEP - TID

Departamento de Engenharia Electrotécnica

Instituto Superior de Engenharia do Porto

7 de Julho de 2017

Agradecimentos

Aproveito esta oportunidade para agradecer a todas as pessoas que contribuíram ao longo desta jornada para a obtenção do grau de mestre.

Em primeiro lugar, gostaria de agradecer ao Eng^{o}. Ricardo Puga (meu orientador) e à Eng^{a} Judite Ferreira (minha co-orientadora) pelo acompanhamento do trabalho, disponibilidade para esclarecimento de dúvidas, pelo material disponibilizado, competência cientifica, assim como críticas, correções, sugestões feitas durante a orientação e pela utilização do laboratório do TID que facilitou o desenvolvimento do projeto.

Aos meus pais e família pelo apoio dado ao longo de todos estes anos.

Aos meus colegas de curso, que de uma maneira ou outra ajudaram, aturaram e acompanharam em grandes noitadas de trabalho durante a elaboração deste projeto de conclusão do percurso académico, e que fizeram toda a diferença.

E por fim gostaria de deixar um especial agradecimento, a todas as pessoas que de uma forma ou outra tiveram o seu contributo neste projeto quer por terem perdido o seu precioso tempo a lerem a tese, distrair-me nos momentos de bloqueio, pela companhia ou simplesmente por ouvir-me a falar "chinês"quando me entusiasmava com o assunto.

A todos, um grande OBRIGADO.

Esta página foi intencionalmente deixada em branco.

Resumo

A medicação intravenosa é algo que requer muitos cuidados e monitorização, tanto no cálculo do fluxo a ser administrado, como na verificação periódica do estado do equipamento. Pretende-se portanto, neste projeto, desenvolver um sistema onde a monitorização da medicação a ser administrada, assim como a verificação de possíveis anomalias do sistema possam posteriormente ser reportadas de modo wireless para uma base de dados, que pode ser gerido a partir de plataforma central de monitorização.

Palavras-chave

Medicação Intravenosa; Monitorização; ZigBee; XBee; Base de Dados; ARM; MySQL; PHP.

Esta página foi intencionalmente deixada em branco.

Abstract

Intravenous medication is something that requires a lot of care and monitoring, both in the calculation of the flow to be administered and the periodic check of the state of the equipment. This project intends to develop a system that monitors the medication to be administered, as well as checking for possible system anomalies, that will subsequently be reported wirelessly to database that can be managed from a central monitoring platform.

Keywords

Intravenous Medication; Monitoring; ZigBee; XBee; Data Base; ARM; MySQL; PHP.

Esta página foi intencionalmente deixada em branco.

Sumário

Este relatório encontra-se dividido em 8 partes:

- *Introdução* - este capítulo será composto por uma contextualização do projeto, exposição dos objetivos e uma breve explicação da organização do documento.
- *Estado da Arte* - nesta parte será feito um estudo dos dispositivos médicos já existentes no mercado para a administração de medicação intravenosa.
- *Especificação Funcional* - neste capítulo será apresentado o sistema idealizado de uma maneira geral.
- *Tecnologia* - este capítulo será destinado ao estudo das alternativas de comunicação entre os elementos do sistema, as técnicas de deteção de gotas, o hardware e as ferramentas de software para o desenvolvimento do projeto.
- *Hardware* - neste capítulo serão expostos os conceitos e teoria aplicada na construção do hardware.
- *Software* - este capítulo será destinado à apresentação dos algoritmos e mecanismos utilizados no decorrer dos diferentes programas.
- *Testes e Resultados* - neste capítulo serão apresentados os resultados dos testes feitos ao sistema em diversas condições e o aspeto final dos elementos do sistema.
- *Conclusões e Trabalho Futuro* - esta parte será destinada à apresentação das considerações finais do projeto e perspetivas futuras para o sistema desenvolvido.

Esta página foi intencionalmente deixada em branco.

Conteúdo

Esta página foi intencionalmente deixada em branco.

Lista de Figuras

Lista de Acrónimos

AMPOP - Amplificador Operacional

ARM - Advanced RISC Machine

BD - Base de Dados

CAN - Control Area Network

DNS - Domain Name System

FFD - Full Function Device

GPIO - General Propose Input/Output

IC - Integrated Circuit

IDE - Integrated Development Environment

JSON - JavaScript Object Notation

PCB - Printed Circuit Board

PHP - Hypertext Preprocessor

RFD - Reduced Function Device

SFTP - Secure File Transfer Protocol

SGBD - Sistema de Gestão de Base de Dados

SPI - Serial Peripheral Interface

SRAM - Static Random Access Memory

TTL - Transistor-Transistor Logic

UART - Universal Asynchronous Receiver/Transmitter

UI - User Interface

USB - Universal Serial Bus

Wi-Fi - Wireless Fidelity

Capítulo 1

Introdução

A expansão das tecnologias *wireless* é algo notório na atualidade e estas encontram-se disseminadas pela grande maioria dos dispositivos móveis. A dimensão e aplicação destas tecnologias faz com que seja quase de uso obrigatório nos sistemas que implicam algum tipo de mobilidade. Algumas das tecnologias mais conhecidas e utilizadas frequentemente são o Wi-fi e Bluetooth. No entanto, muitas outras tecnologias existem no mercado, como é o caso do ZigBee, ANT+, Z-Wave, RFID, etc.

Com o aparecimento destas novas tecnologias, também se criaram novas funcionalidades, sendo que, atualmente, as redes sem fios não são apenas utilizadas para transferir ficheiros e navegar na internet. Hoje em dia podem-se encontrar tecnologias *wireless* em vários domínios, como na domótica, na automação, no fitness, nos cuidados de saúde e até em domínios onde já eram utilizadas tecnologias rádio, substituindo assim estas.

Na área dos cuidados de saúde, as tecnologias *wireless* podem dar um importante contributo para que sejam melhoradas as condições de monitorização, qualidade de serviço, tratamento e gestão mais eficiente dos recursos dos profissionais de saúde. Com a utilização deste tipo de tecnologias, é possível fazer um registo mais autónomo e, em caso de deteção de anomalia, lançar um alarme direto aos responsáveis para uma resposta mais rápida e

eficiente.

1.1 Contextualização

Este projeto surge como tese/dissertação com o objetivo de terminar o ciclo de estudos desenvolvidos ao longo dos últimos anos. Após analisar a lista de projetos viáveis para a execução deste, a escolha foi direcionada para a área da eletro-medicina.

A opção deste projeto foi feita tendo em conta o gosto, desejo e curiosidade de conseguir aliar a eletrónica à medicina de maneira a poder desenvolver um dispositivo de aplicação médica.

Dentro da área da saúde existem várias zonas de aplicação de comunicação *wireless*, no entanto foi escolhida a monitorização de medicação intravenosa por ser, de certa forma, uma área pouco desenvolvida e a tecnologia utilizada ou é a tradicional (onde não há nenhum tipo de monitorização constante) ou então existem aparelhos sofisticados (no entanto o custo é bastante elevado). Ao longo deste projeto pretendeu-se desenvolver uma solução que fosse acessível a mais pessoas e que permitisse desta forma um aumento de qualidade do tratamento, monitorização correta da medicação e deteção de anomalias.

1.2 Objetivos

O principal objetivo deste projeto, é a construção de um sistema capaz de fazer a monitorização e registo wireless, dos valores de medicação administrados pela via intravenosa. No entanto, dada a complexidade inerente a este único objetivo, sentiu-se a necessidade de fazer a divisão deste trabalho em tarefas de dimensões mais reduzidas, nomeadamente:

- Investigação do tipo de aparelhos semelhantes já existentes no mercado;

- Estudo de tecnologias de comunicação wireless do sistema;
- Estudo de técnicas para a deteção da medicação;
- Implementação de uma das técnicas estudadas;
- Desenvolvimento do módulo de deteção;
- Desenvolvimento de mecanismo para registo dos dados;
- Criação de uma plataforma de consulta e gestão dos registos.

1.3 Organização do Relatório

No Capítulo 1, denominado "Introdução", pretende-se fazer o enquadramento do projeto, um breve resumo do mesmo, definindo quais os objetivos que se pretendem atingir e, por fim, apresentar de forma resumida o que é tratado em cada capítulo.

No Capítulo 2, chamado "Estado da Arte", pretende-se fazer uma apresentação dos modos de administração de medicamentos possíveis, assim como mostrar os dispositivos médicos já existentes no mercado para o mesmo tipo de aplicação. Este capítulo contém também uma pequena comparação dos dispositivos, salientando as vantagens e desvantagens de cada um deles.

No Capítulo 3, designado "Especificação Funcional", pretende-se expor a arquitetura do sistema idealizado, assim como explicar os diferentes módulos que constituem o sistema e o papel de cada um.

No Capítulo 4, intitulado "Tecnologia", tem-se como objetivo fazer o levantamento e posterior seleção das tecnologias utilizadas neste projeto para a comunicação entre os diferentes módulos, fundamentos teóricos para a medição de gotas e as ferramentas/serviços utilizados.

No Capítulo 5, nomeado "Hardware", são explicados todos os fundamentos utilizados para o desenvolvimento do hardware das diferente partes do projeto.

No Capítulo 6, denominado "Software", é feita a explicação do papel de cada um dos softwares desenvolvidos para este sistema, assim como os algoritmos e raciocínios por trás destes.

No Capítulo 7, chamado "Testes e Resultados", pretende-se fazer uma apresentação do resultado final do funcionamento do sistema.

No ultimo capítulo, designado "Conclusões e Trabalho Futuro", são reunidas as principais conclusões e é feita uma perspetiva de possíveis desenvolvimentos futuros.

Capítulo 2

Estado da Arte

Ao longo deste capítulo será apresentado ao leitor uma visão geral dos modos de administração possíveis da medicação, focando posteriormente alguns factos sobre a administração pela via intravenosa, exposição dos sistemas de infusão disponíveis no mercado, assim como as suas características, finalizando com uma breve conclusão do tema e fazendo uma introdução o próximo capítulo.

O Estatuto do Medicamento, Decreto-Lei 76/2006 de 30 Agosto, transpõe esta definição comunitária para o contexto nacional. Segundo o Decreto-Lei 76/2006 de 30 Agosto, "≪Medicamento≫, toda a substância ou associação de substâncias apresentada como possuindo propriedades curativas ou preventivas de doenças em seres humanos ou dos seus sintomas ou que possa ser utilizada ou administrada no ser humano com vista a estabelecer um diagnóstico médico ou, exercendo uma ação farmacológica, imunológica ou metabólica, a restaurar, corrigir ou modificar funções fisiológicas".[8]

A administração de medicação depende, por exemplo, da rapidez de atuação, tipo de medicação, quantidade da mesma a ser administrada e condições do paciente. Tendo em conta os fatores apresentados anteriormente, podemos constatar que a aplicação de um fármaco a um paciente pode apresentar limitações no que toca à via utilizada para a administração

pois, dependendo deste, esta pode ser contra-indicada em alguns casos.

A **via de administração** pode ser definida como o caminho pelo qual a droga é colocada no organismo. Cada uma das diferentes vias disponíveis apresenta objetivos específicos, assim como vantagens e desvantagens. As diferentes vias são:

- *Oral* - sendo ingerida pela boca;
- *Endovenosa/Intravenosa* - injetando numa veia;
- *Intramuscular* - injetando num músculo;
- *Ocular* - instilando no olho;
- *Nasal* - vaporizando nas fossas nasais;
- *Inalação* - vaporiza na boca;
- *Tópico* - aplicando sobre a pele;
- *Transdérmico* - sistémico.

O sistema circulatório é o principal caminho para a distribuição de oxigénio e outros nutrientes pelo corpo sendo também a mais rápida, uma vez que um ciclo completo para o suprimento de todo o corpo dá-se em 60 segundos, o que ajuda a perceber o porquê da via intravenosa ser utilizada por aproximadamente 80% dos pacientes hospitalizados. Esta via permite um meio eficiente no que toca ao fornecimento de sangue, medicamentos e fluído para os órgãos vitais dos pacientes.[2]

A administração de medicação pela via intravenosa pode ser feita de duas maneiras distintas:

- *Infusão Contínua* - é administrada ao longo de várias horas, com recurso ao gotejamento contínuo;

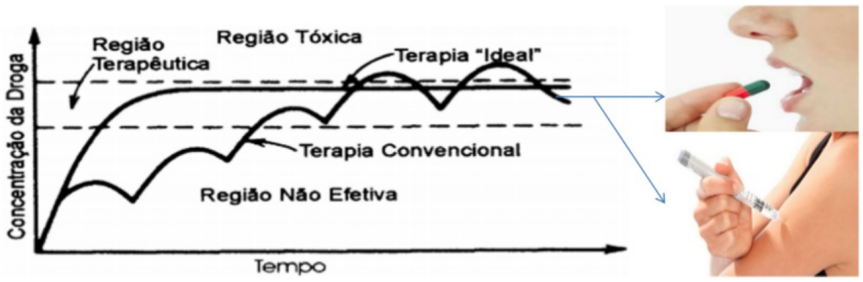


Figura 2.1: Concentração da medicação ao longo do tempo com terapia convencional.[1]

- *Infusão Intermitente* - é introduzida no sistema de um paciente num curto intervalo de tempo. Pode ser administrada:
  - Em *Bolus* - a substância (não diluída) é administrada toda de uma vez.
  - *Infusão Secundária* - o fármaco é diluído num pequeno volume de solução (50 a 100ml) durante um curto espaço de tempo (entre 30 a 60 minutos).

Quando uma medicação é dada a um paciente, através de pílulas ou injeções, observa-se flutuação de concentração da substância no organismo do paciente. Este tipo de flutuação faz com que, ao longo do tempo a concentração da droga aplicada possa oscilar entre a região não efetiva e a tóxica, tal como se pode verificar na Figura 2.1. A infusão contínua do medicamento permite que, com uma taxa de infusão correta, manter a concentração da medicação na zona terapêutica [1]

Um sistema de infusão, deve regular a concentração do fármaco no corpo do paciente, de modo a alcançar o resultado esperado. Uma vez que a monitorização direta da concentração da medicação não é de fácil verificação, é normalmente assumida uma taxa de infusão ou concentração específica sanguínea que permite alcançar o objetivo farmacêutico da droga. A concentração da substância terapêutica aplicada ao paciente deve ser respeitada,

pois se esta estiver abaixo da necessária não provoca o efeito pretendido e, por outro lado, se estiver acima pode ter efeitos tóxicos no paciente. Estes riscos, assim como a taxa a ser administrada, dependem não só do fármaco, mas também do paciente. [2]

A administração de fluídos e eletrólitos pela via intravenosa normalmente não requer uma regulação de grande precisão, podendo mesmo em casos de baixo risco admitir um erro na taxa de infusão de cerca de 30% dos fluídos. Existem casos e certos medicamentos (especialmente agentes cardioativos fortes) que requerem alta precisão. [2]

A infusão intravenosa é um procedimento que é feito de forma prolongada e, especificamente em doentes com restrição de fluídos. Uma administração acima ou abaixo da região terapêutica pode comprometer os sistemas renais e cardiovasculares do paciente. [1]

2.1 Sistemas de Infusão

Os sistemas de infusão intravenosa tornaram-se uma opção de escolha para o fornecimento de um grande tipo de fluídos e fármacos tanto em ambientes hospitalares como noutros de cuidados de saúde. Estes dispositivos oferecem uma vasta gama de opções para fazer a infusão. A existência de vários tipos de sistemas, implica que o técnico se saúde seja forçado a ter conhecimento para poder decidir qual o mais apropriado para a aplicação, uma vez que cada um apresenta propriedades específicas. Esta decisão baseia-se no conhecimento da mecânica de fluídos, farmacologia e propriedades do medicamento. A evolução contínua do desempenho, fiabilidade, segurança e custo destes tipos de sistemas permite, cada vez mais, uma amplitude maior de utilização de infusão intravenosa em diversos casos. [2]

Os sistemas de infusão intravenosos têm tipicamente três **componentes** que podem ser dados como principais, sendo eles: o reservatório de fluídos, sistema de cateter e dispositivo de regulação e/ou geração de fluxo.[2]

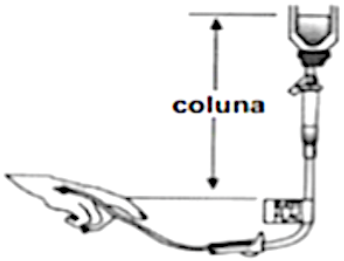


Figura 2.2: Coluna de Líquido[1]

Os dispositivos de infusão devem ser tidos como uma ajuda para os profissionais de saúde e não como uma substituição ou alívio das suas responsabilidades. Como tal, os profissionais devem periodicamente monitorizar e documentar as taxas de administração prescritas para o tratamento.[9]

2.1.1 Equipamento de Controlo de Fluxo Manual

O sistema de controlo por fluxo manual é considerado o mais simples. A sua taxa de infusão é regulada manualmente por um operador, garantindo assim, que os valores prescritos são respeitados. Este tipo de controlo de infusão apenas deve ser usado para regular infusões simples que apresentem um baixo risco. Aquando da seleção do tipo dispositivo de infusão deve-se ter em consideração a idade e condição do paciente, o tratamento prescrito e também as condições em que é aplicado. [9]

A pressão de infusão depende do tamanho da coluna de líquido (Figura 2.2) e da pressão venosa do paciente. O fluxo gerado por gravidade não pode ser utilizado em infusões arteriais, uma vez que a pressão vascular é superior à pressão hidrostática gerada pelo equipamento.[2]

Este tipo de sistema é constituído por três componentes (Figura 2.3): frasco de fluído (reservatório que contém o líquido de infusão), câmara de gotejamento (câmara que permite monitorizar a taxa de infusão a ser administrada) e o rolete (permite comprimir o tubo do sistema para controlar o fluxo do líquido a fornecer ao paciente).

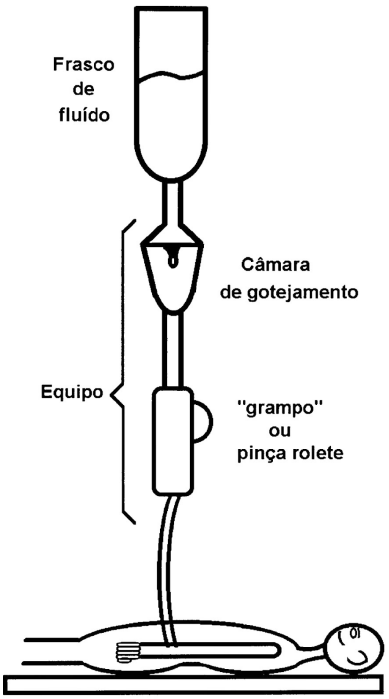


Figura 2.3: Elementos do Controlo de Fluxo Manual.[1]

Este tipo de sistema apresenta como **vantagens**:[1]

- baixo custo;
- simples na operação.

No entanto apresenta também algumas **desvantagens** tais como:[1]

- imprecisão de medição;
- variação de fluxo ao longo do tempo;
- redução da coluna de líquido ao longo do tempo provoca alterações na taxa de infusão;
- variações da pressão venosa;
- depende da altura do reservatório;
- viscosidade e temperatura do líquido;

Figura 2.4: Controlador Automático de Infusão.

- não possui qualquer tipo de alarme (monitorização apenas feita pelo profissional de saúde);

- apenas pode ser aplicado em veias periféricas onde a pressão sanguínea é baixa.

2.1.2 Controlador de Infusão

O controlador de infusão permite, em relação ao sistema anterior, um maior controlo do fluxo. Este sistema é capaz de regular a taxa de medicação aplicada ao paciente, através da contagem eletrónica de gotas [1, 2]. O controlo da administração de medicamentos deste sistema pode ser feito de duas maneiras distintas, nomeadamente através do sistema de infusão por gravidade com controlador automático ou do sistema de infusão por gravidade com controlador semiautomático.[1]

No **sistema de infusão por gravidade com controlador automático** (Figura 2.4) quando o fluxo de administração do fluído está fora dos parâmetros, este pode atuar sobre o rolete, de maneira a regular o fluxo para o valor pré-determinado pelo profissional de saúde[1]

No **sistema de infusão por gravidade com controlador semiautomático**(Figura 2.5) quando o fluxo de administração do fluído está fora dos parâmetros, este emite um alarme sonoro para avisar o operador que deve ajustar o rolete para que a infusão volte aos valores desejados.[1]

Figura 2.5: Controlador Semiautomático

Este tipo de sistema apresenta como **vantagens**:[1]

- baixo custo;
- controlo dos valores de medicação.

No entanto apresenta também algumas **desvantagens** tais como:[1]

- apenas pode ser aplicado em veias periféricas onde a pressão sanguínea é baixa.
- sensíveis a oclusão do equipamento;
- sensíveis ao deslocamento da agulha.

2.1.3 Bomba de Infusão

A bomba de infusão permite fazer a administração de medicação com maior precisão e segurança durante longos períodos de tempo ou onde são necessários maiores fluxos. Este mecanismo de administração permite também outras vantagens como: a aplicação em pacientes de medicação em que o volume total não pode ser ultrapassado, representa um modo efetivo para a segurança do paciente e permite também fazer terapia intra-arterial uma vez que existe pressão positiva suficiente para vencer a pressão do vaso sanguíneo. De um modo geral, este sistema caracteriza-se por não depender

da pressão gravitacional, pois combina eletrónica com mecanismos de infusão (peristáltico, por pistão ou seringa) permitindo fazer o controlo tanto volumétrico como não-volumétrico.[1, 2, 9]

Por razões de segurança para o paciente, o equipamento a ser selecionado para uma determinada aplicação deve ter em consideração as especificações de segurança do próprio aparelho. Estas incluem (mas não são limitadas a) alarmes sonoros, autonomia da bateria, indicações de operação, proteções contra fluxo livre, mecanismo ajustável de pressão de oclusão, precisão da administração, cálculo de dosagem, monitorização da pressão no sistema e mecanismo de prevenção de manipulação de dados.[9]

As bombas de infusão são compostas por diversos elementos, nomeadamente[1]:

- **Circuito de Controlo** - circuito responsável por fazer o controlo da dosagem da medicação;
- **Sensor da gotejamento** - deteta a presença de gotas e a sua frequência;
- **Sensor de Ar** - deteta bolhas de ar no sistema de soro do paciente e envia sinal de bloqueio de infusão de modo a evitar embolia;
- **Painel de Controlo** - normalmente consiste num teclado que permite a programação dos parâmetros de administração;
- **Display** - apresenta dados da administração em execução;
- **Alarmes** - indicam anomalias à operação esperada;
- **Motores** - usualmente motores passo a passo são utilizados para sistemas de infusão pois são capazes de executar movimentos com grande precisão;
- **Mecanismo de Infusão** - permite regular o fluxo de medicação a ser dado ao paciente.

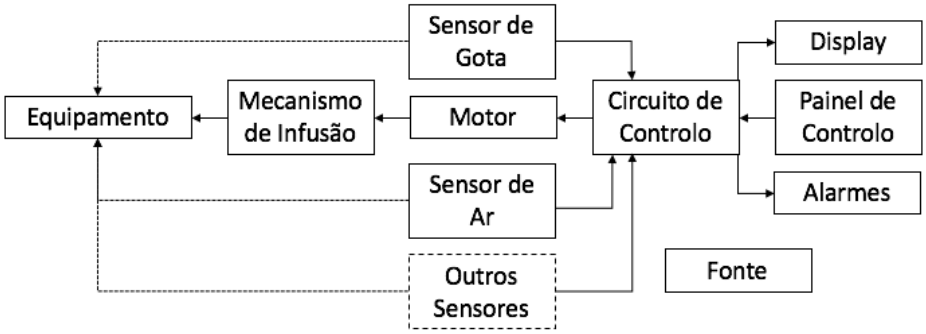


Figura 2.6: Diagrama geral de blocos dos constituintes de uma bomba de infusão.

O **Circuito de Controlo**, que permite fazer a monitorização da medicação aplicada num determinado momento, pode ser classificado como controlo volumétrico e não-volumétrico dependendo da metodologia que é utilizada para fazer a medição. O *Controlo Volumétrico* faz a monitorização pelo fluxo de líquido infundido (unidade é ml/h), fazendo o cálculo a partir da velocidade de infusão. Este modo tem a vantagem de não ser dependente das características do líquido. O *Controlo Não-Volumétrico* faz a monitorização a partir da quantidade de gotas libertadas pelo gotejador conseguindo, a partir delas, controlar a velocidade de infusão. Este último depende do volume da gota (microgota/macrogota) e deve ser tido em atenção o tipo de equipamento utilizado, a temperatura, a viscosidade e densidade do líquido.[1]

Os **Mecanismos de Infusão** permitem regular o fluxo de medicação a ser dado ao paciente. A metodologia utilizada para fazer esta regulação pode ser feita de três diferentes técnicas, designadamente, por efeito peristáltico, por seringa ou pistão. [1]

O mecanismo Peristáltico tem a sua base de funcionamento no esmagamento de uma pequena porção do tubo do equipamento (Figura 2.8). Este tipo de mecanismo permite variar o fluxo entre 0,01 e 1000ml/h e o volume máximo que pode ser administrado está limitado, apenas, pela capacidade

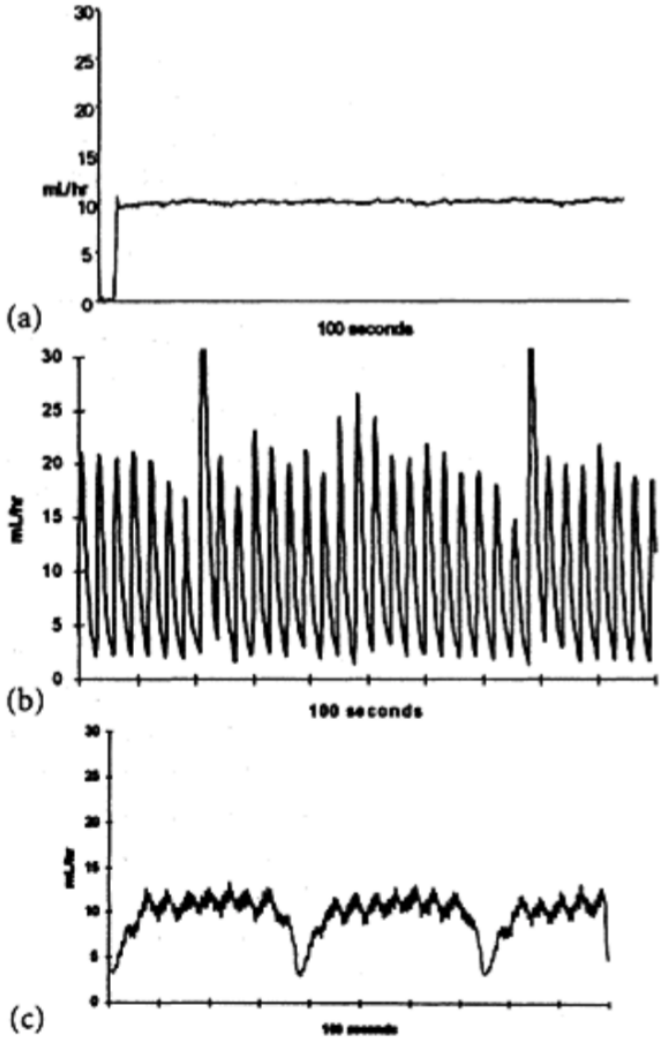


Figura 2.7: Gráfico representativo de administração contínua de 10mL/h durante 100 segundos para os mecanismos: (a) Seringa, (b) Pistão e (c) Peristáltico Linear [2]

do reservatório. Os mecanismos peristálticos embora funcionem com base no mesmo princípio podem ser divididos em dois subtipos, dependendo do método utilizado, nomeadamente, como lineares ou rotacionais. [1, 2]

O mecanismo *Peristáltico Linear* (Figura 2.8a) pressiona o segmento do tubo contra uma superfície rígida. Para pressionar o tubo é utilizado um conjunto de engrenagens em série que vão sequencialmente fazendo a oclusão do tubo desde o extremo mais próximo do reservatório até ao outro extremo mais próximo do paciente. Normalmente este tipo de mecanismos possui um motor passo a passo, que vai incrementando o passo a um intervalo constante, o que resulta numa administração que segue uma flutuação semelhante à apresentada na Figura 2.7c. [2]

O mecanismo *Peristáltico Rotacional* (Figura 2.8b) atua sobre uma pequena porção do tubo do equipamento. A sua base de funcionamento reside na compressão do tubo contra um rolamento situado na extremidade do rotor do motor. A rotação do rotor faz com que os rolamentos empurrem o fluído do reservatório pelo tubo até ao paciente. Durante a rotação do rotor, pelo menos um rolamento deve garantir a oclusão total do tubo de maneira a evitar que haja fluxo livre do líquido para o paciente. No decurso do ciclo de rotação haverá, pelo menos, dois rolamentos a bloquear o fluxo num segmento do tubo, sendo que o volume que fica entre os dois rolamentos vai determinar a precisão volumétrica de administração do dispositivo. [2]

O mecanismo de infusão do tipo Seringa (Figura 2.9) permite a administração de um fluxo contínuo de alta precisão (podendo ser inferior a 1% de erro) para pequenos volumes, devido ao volume da seringa (<100ml). Normalmente, este tipo de mecanismo é utilizado na infusão de medicamentos que apresentam uma elevada concentração por elevados períodos de tempo. A precisão da administração de medicamentos concentrados é garantida a partir de um motor passo a passo com redução de velocidade, que aciona o êmbolo da seringa resultando numa administração semelhante ao da Figura

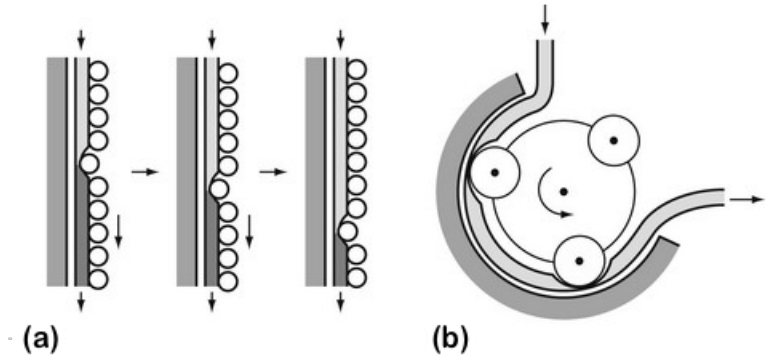


Figura 2.8: Mecanismo de infusão Peristáltico (a)Linear e (b)Rotativo[3]

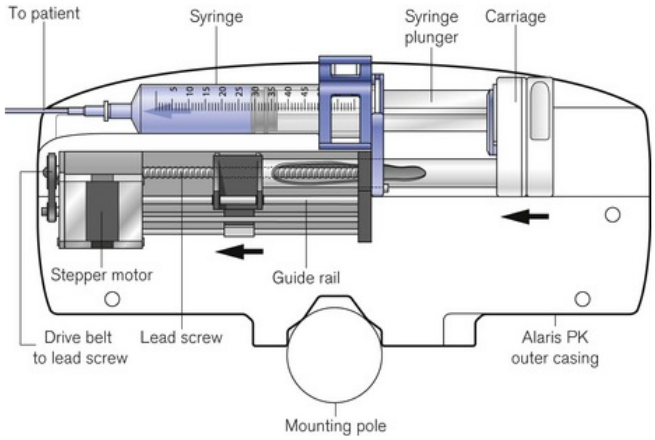


Figura 2.9: Mecanismo de Infusão por Seringa.[3]

2.7a. Alguns modelos possibilitam a montagem de múltiplas seringas, o que permite tanto a aplicação contínua (quando uma das seringas termina), como também infusões simultâneas de vários medicamentos. [1, 2]

O fluxo de administração depende não só da velocidade do êmbolo, mas também do diâmetro da seringa, sendo necessário os fabricantes especificarem os tipos e tamanhos das seringas perfusoras a serem aplicados em cada um dos modelos de modo a evitar um fluxo incorreto e problemas de alarmes. Os alarmes presentes permitem chamar a atenção do operador para situações anómalas que podem ser perigosas para o paciente tais como seringa vazia (fim da infusão), alta pressão na seringa (oclusão), bateria des-

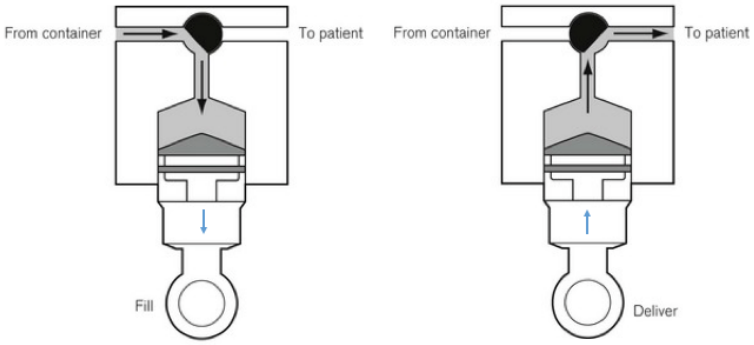


Figura 2.10: Mecanismo de Infusão por Pistão.[3]

carregada e mau funcionamento da bomba. Por outro lado, ao contrário de outros mecanismos, este não apresenta sensores de ar na linha dada a baixa probabilidade de ser infundido ar no paciente, devido aos baixos volumes e fluxos existentes neste tipo de mecanismo.[1, 2]

O mecanismo de infusão por Pistão ou Cassete (Figura 2.10) utiliza um equipamento que é constituído por um motor, uma câmara, uma válvula, um tubo e um pistão. Quando o motor roda, o pistão entra e sai do êmbolo fazendo com que a câmara se encha e esvazie de fluído do reservatório. A válvula é responsável por fazer o direcionamento do fluxo conforme o ponto do ciclo em que se encontra do bombeamento. Nos mecanismos que possuem apenas um pistão pode-se verificar (Figura 2.7b) que existe uma grande flutuação de fluxo, uma vez que o processo de enchimento da câmara requer uma interrupção do fluxo de metade do ciclo, isto reflete-se na estabilidade da administração. Para contornar este problema alguns dispositivos desta categoria fazem uso de dois pistões desfasados de modo a proporcionar um fluxo praticamente constante. [1, 2]

2.2 Conclusão

A administração de medicação progrediu de maneira a melhorar a eficiência desta e a segurança dos pacientes. Para tal, assiste-se à diversa oferta que

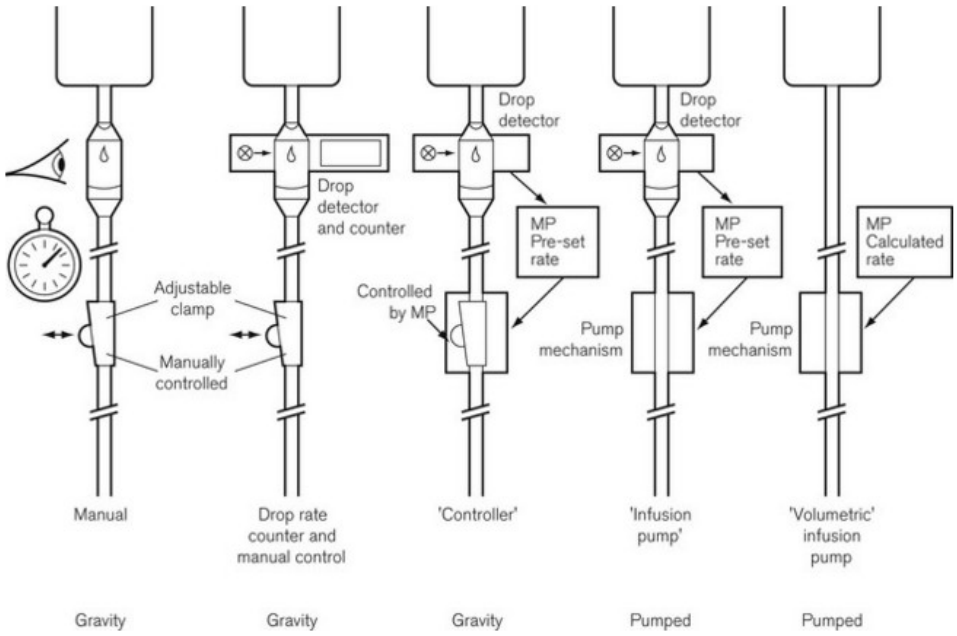


Figura 2.11: Evolução dos dispositivos de infusão.[3]

o mercado apresenta, desde dispositivos que não possuem qualquer tipo de monitorização e de ajuste manual, até dispositivos tecnológicos de alta precisão capazes de fazer a deteção de falhas, a regulação da dosagem de maneira a garantir a correta concentração de medicamento no paciente e evitar situações que possam impactar negativamente com o estado de saúde do paciente (Figura 2.11).

Apesar da tecnologia ter avançado e de existirem dispositivos de alta precisão, como as bombas de infusão, o número de incidentes neste tipo de aparelhos está longe de ser inexistente. Muitos dos incidentes tem como causa o software, especialmente no que toca ás bombas volumétricas e de seringa (Figura 2.12a). Outro fator que deve ser tido em atenção é que a maioria dos incidentes neste tipo de dispositivos, pode em grande parte, ter consequências muito graves podendo mesmo causar o óbito de um paciente (Figura 2.12b).[4]

Para além de não haver sistemas infalíveis e o custo de equipamentos

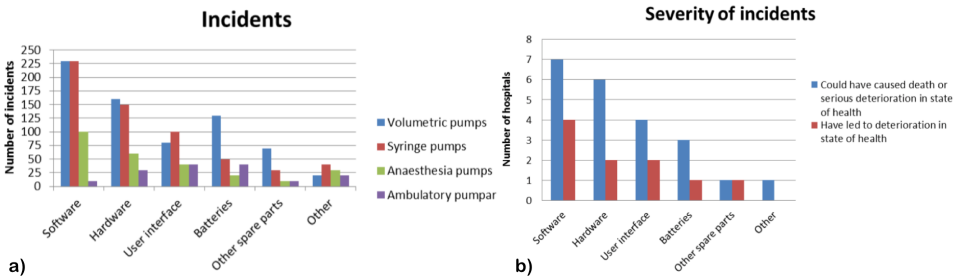


Figura 2.12: a) Incidentes com bombas de infusão; b) Gravidade dos incidentes com bombas de infusão.[4]

mais sofisticados ser elevado, existe também a questão que se prende com a monitorização dos dispositivos. Esta, apesar da fiabilidade e sofisticação do aparelho, depende do fator humano que deve periodicamente verificar e documentar o processo de infusão (monitorizar o fluxo apropriado e correção dos procedimentos em caso de alarme ou anomalia) [9]. A aliar a estes fatores temos ainda de ter em consideração a quantidade de camas que uma instituição pode possuir (que pode chegar a aproximadamente 900 em grandes instituições[4]), e a falta de informação centralizada e em tempo real por parte dos aparelhos.

No sentido de tentar auxiliar as instituições e os próprios profissionais a oferecer uma melhor prestação de serviços para o paciente, surgiu este projeto onde se pretende desenvolver um sistema de infusão que tem como objetivo ser economicamente mais viável que algumas soluções mais sofisticadas e centralizar a informação em tempo real de modo a que seja possível canalizar de maneira mais eficaz o trabalho dos profissionais de saúde. No próximo capítulo será apresentado a estrutura geral que o projeto irá seguir de maneira a tentar satisfazer os objetivos.

Capítulo 3

Especificação Funcional

Neste capítulo pretende-se expor a arquitetura do sistema idealizado. Para tal, primeiro serão apresentados os objetivos que se pretendem atingir de maneira geral, de seguida serão descritos os elementos que vão constituir o sistema e finalmente será elaborado um diagrama geral onde se pretende ilustrar qual será a relação entre os diferentes elementos.

Durante este projeto pretende-se desenvolver um sistema de infusão que apresente a informação centralizada (sem perder a mobilidade) e permita alertar o responsável para a administração de medicação em caso de valores anómalos. Para tal, pode-se dizer que existem alguns objetivos gerais que se pretende atingir, nomeadamente:

- fazer a monitorização da medicação;
- criação de uma rede wireless que permita mobilidade dos dispositivos;
- estabelecer comunicação *wireless* entre dois pontos da rede;
- registo de dados de monitorização numa base de dados para permitir constituir um histórico do paciente;
- registar o valor prescrito;

- alertar o responsável quando os valores se encontrarem fora dos limites estabelecidos para o paciente.

De modo a poder atingir todos estes objetivos foi necessário desenvolver uma arquitetura que pudesse suportar toda esta infraestrutura. Para tal pretende-se dividi-la em pequenos blocos mais simples, de forma a que se complementem e funcionem em conjunto. Tendo isso em mente, o projeto foi idealizado para seguir a estrutura presente na Figura 3.1, que apresenta os seguintes elementos:

- *Base de Dados (BD)* - armazena todos os registos;
- *Servidor* - faz a interface com a BD e responde a pedidos do exterior;
- *Interface com Utilizador (UI)* - permite ao utilizador do sistema aceder aos dados que estão na BD e gerir alguns conteúdos.
- *Router* - faz a inserção na BD (através do Servidor) de registos de infusão momentâneos vindos do(s) monitor(es) de infusão a ele ligado(s).
- *Monitor de infusão* - faz a monitorização da taxa de administração e periodicamente envia o estado atual para o Router.

Uma vez feita a especificação do sistema idealizado, é necessário proceder à pesquisa das ofertas de mercado para saber quais as tecnologias existentes de hardware, software e os protocolos de comunicação. Posteriormente, será feita a seleção das tecnologias que se pretendem utilizar de entre as diferentes tecnologias para cada um dos casos.

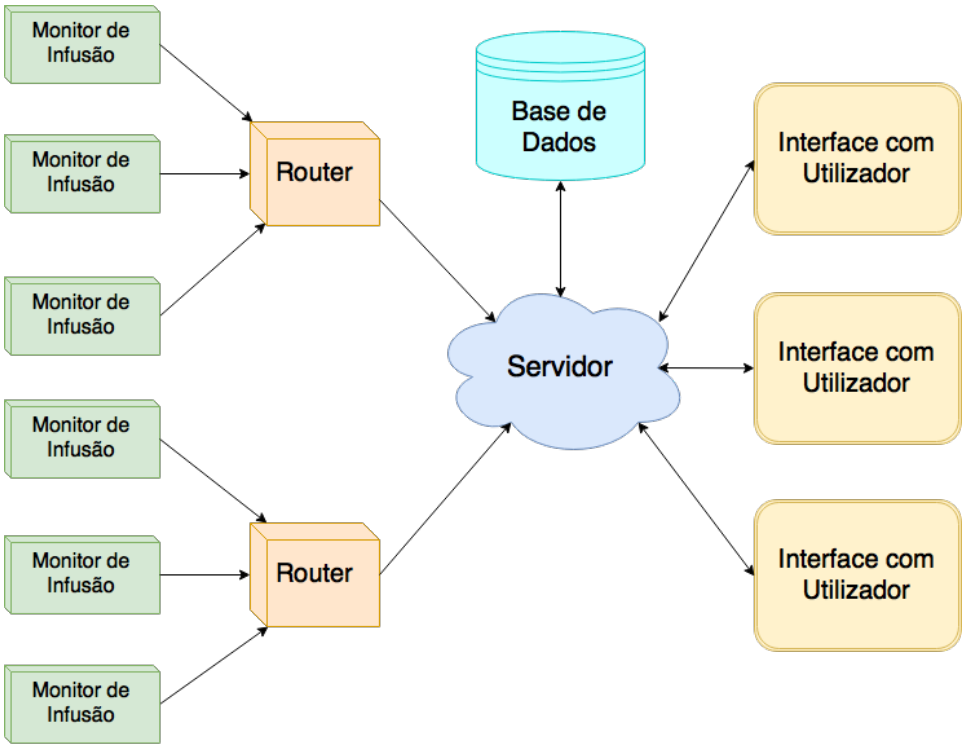


Figura 3.1: Diagrama Geral do Sistema

Esta página foi intencionalmente deixada em branco.

Capítulo 4

Tecnologia

Após ser dada uma breve explicação de como o sistema foi pensado (no Capítulo 3), passamos para uma análise mais aprofundada da tecnologia necessária para poder proceder posteriormente ao seu desenvolvimento. Neste capítulo pretende-se abordar quais são os protocolos de comunicações, hardware e software disponíveis no mercado e as decisões tomadas quanto à utilização, ou não, de cada um deles.

4.1 Protocolo de Comunicação

No capítulo anterior foi apresentada a arquitetura em que o sistema se baseou, assim como os blocos que o constitui. Para haver interação entre os diferentes elementos é necessário estabelecer comunicação entre eles.

Primeiramente é preciso entender as necessidades de cada uma das partes constituintes, sendo que, neste caso podemos de uma maneira não muito aprofundada dizer que, entre o Módulo de Infusão e o Router é essencial garantir mobilidade e, como tal, a comunicação deverá ser *wireless*, de baixo consumo energético (pois poderá apresentar uma bateria e assim melhorar a autonomia do dispositivo) e apresentar uma baixa taxa de transmissão pois, os dados não deverão ser de alto volume e apenas será necessário,

periodicamente, atualizar os dados.

No que toca às comunicações entre o Router e o Servidor deverá garantir-se que a comunicação pode suportar altas taxas de informação uma vez que o Router é considerado um nó e, desta forma, o tráfego será tanto maior quanto o número de dispositivos ligados a ele. No entanto, não será necessário garantir que seja wireless, nem de tão baixo consumo como os Módulos de Infusão, uma vez que estes dispositivos fazem parte da infraestrutura e, como tal, não é problemática a questão da mobilidade (ser *wireless*) nem o consumo energético (uma vez que deve estar permanentemente ligado à corrente elétrica).

Quanto à comunicação, entre o Interface com o Utilizador e o Servidor, deverá ser bastante semelhante com a que foi definida entre o Router e o Servidor, podendo apenas ser aberta uma questão quanto à possível mobilidade dos equipamentos (requerendo por isso wireless e alguma preocupação quanto ao consumo energético). No que toca à taxa de dados transferidos poderá ser elevada, dependendo da quantidade e dos pedidos em si, que possam fazer ao servidor.

4.1.1 Redes Sem Fios e o Enquadramento na Norma IEEE

A normativa que define o standard das redes sem fios ou redes *wireless* é a norma IEEE 802. Esta norma define as especificações para a camada física e para camada controlo no acesso ao meio numa rede sem fios.

Conforme as características da rede, pode ser dividida em quatro grupos [10, 11]:

- ***Wireless Local Area Network* (WLAN)** - Agrupa as tecnologias com alcance entre os 100m e os 300m, normalmente como extensão de redes com fios convencionais;

- ***Wireless Metropolitan Area Network* (WMAN)** - Engloba os

acessos por banda larga em redes metropolitanas, com alcance em torno dos 6km;

- ***Wireless Wide Area Network* (WWAN)** - É o grupo mais amplo em termos de alcance e é orientado para serviços de telecomunicações de longa distância.

Algumas Tecnologias Sem Fios

Nas últimas décadas a automatização industrial tem sofrido um grande desenvolvimento, caracterizado por incorporar diferentes áreas modernas tais como: a comunicação, a informação, a computação, o controlo, a aquisição de dados e a atuação de uma maneira totalmente integrada. Deste processo resultam novas soluções, com melhor eficiência e sistemas completos. Uma das componentes que tem adquirido uma importância maior na indústria é a comunicação. Para existir interligação entre os sistemas automatizados, estes podem ser integrados com vários sensores, controladores e máquinas que diferem entre si nas funcionalidades, mas que apresentam uma especificação comum para a troca de mensagens. Muitos tipos de rede diferentes foram indicadas para o uso em chão de fábrica, incluindo CAN (*Control Area Network*), Profibus (*Process fieldbus*), Modbus, etc. Contudo, a maneira de selecionar um tipo de rede adequada para uma aplicação em particular é um problema crítico no que toca à engenharia industrial.[12]

Por outro lado, existe um mercado crescente de tecnologias sem fios, que proporcionam uma maior mobilidade e flexibilidade dos sistemas, apresentando como principal vantagem o facto de não ter cabos, em relação aos dispositivos tradicionais. Os benefícios deste tipo de tecnologias não se restringem à inexistência de cabos, mas também com a redução de custo, a formação dinâmica de uma rede e a fácil instalação.[12]

No que toca às redes sem fios de curto alcance, podemos afirmar que existem atualmente quatro tecnologias standards que se afirmam: Bluetooth

(IEEE 802.15.1), *UWB* (IEEE 802.15.3), ZigBee (IEEE 802.15.4) e *Wi-Fi* (IEEE 802.11a/b/g). A norma IEEE é responsável pela definição das camadas físicas (PHY) e MAC para as comunicações *wireless* num raio de ação de 10-100 metros. [12]

Baker em [13] estudou as vantagens e desvantagens dos protocolos ZigBee e Bluetooth para aplicações industriais. No seu estudo chegou à conclusão que, o ZigBee pode abranger uma maior variedade de aplicações comparativamente ao Bluetooth uma vez que, garante uma maior autonomia de bateria, maior alcance, flexibilidade (no que toca à dimensão da rede) e fiabilidade, dado a sua arquitetura de rede *mesh*.[12]

O IEEE apenas define as camadas PHY e MAC das normas. Todas as restantes (rede, segurança e aplicação) são desenvolvidas por parcerias entre empresas que trabalham em conjunto com o objetivo de aumentar o potencial comercial que a norma desenvolvida possa vir a ter.[12]

O **Bluetooth** (IEEE 802.15.1) é baseado num sistema de rádio pensado para dispositivos baratos e de curto alcance, com o intuito de substituir os cabos nos periféricos dos computadores (rato, teclado, *joystick*, impressora, etc.). Este tipo de aplicações insere-se dentro das WPAN (*Wireless Personal Area Network*).[12]

Existem duas topologias de conexão definidas para o Bluetooth[12]:

- *Piconet* - é uma WPAN formada por um dispositivo Bluetooth que serve de *master* (Figura 4.1 dispositivos M) e outro(s) que servem de *slaves*(Figura 4.1 dispositivos S). As comunicações neste tipo de topologia apenas se processam em modo *point-to-point* segundo ordem do *master*, no entanto este pode comunicar tanto em modo *point-to-point* ou *point-to-multipoint*. Todos os dispositivos na *piconet* são sincronizados pelo relógio do *master*. Para reduzir o consumo energético, os *slaves* podem estar em modo *parked* ou *standby*.

- *Scatternet* - é um conjunto operacional de *piconets* que se sobrepõe

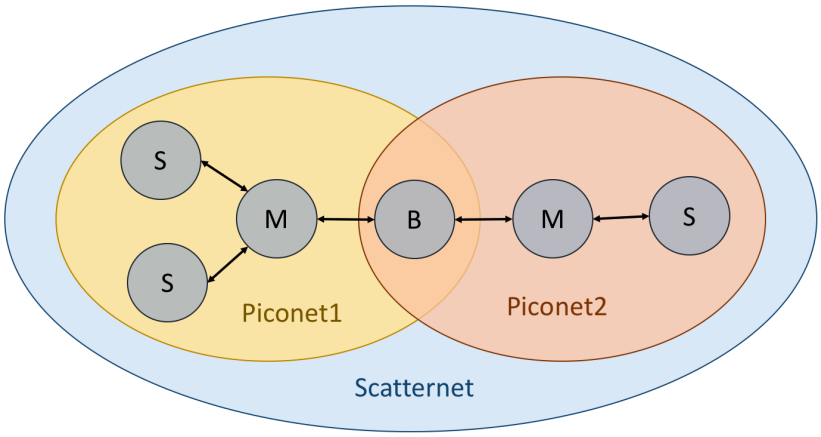


Figura 4.1: Topologias de comunicação Bluetooth

no tempo e no espaço. Um dispositivo pode pertencer a múltiplas *piconets*(Figura 4.1 dispositivo B), permitindo que a informação possa fluir entre *piconets*. Um dispositivo pode desempenhar o papel de *slave* a várias *piconets*, no entanto, apenas pode ser *master* em uma delas.

Entretanto, há uma protocolo, o UWB, que chama a atenção por ser uma comunicação *wireless* de alta velocidade e de curto alcance. Uma das principais características do UWB é a sua largura de banda acima dos 110Mbps (até 480Mbps) que permite satisfazer a maioria das aplicações multimédia, tal como a transferência de áudio e vídeo numa rede doméstica. Pode também funcionar como substituto do cabo em barramentos de alta velocidade como por exemplo o USB 2.0 e o IEEE 1394.[12]

O **Wi-Fi** (*Wireless Fidelity*) inclui as normas IEEE 802.11a/b/g para WLAN (*Wireless Local Area Network*). Isto permite que os utilizadores naveguem na *internet* a altas velocidades quando conectados a AP (*Access Point*) ou interligados entre si, sem necessidade de um *router* (modo *ad hoc*). A arquitetura IEEE 802.11 consiste na interação de vários componentes, para proporcionar uma rede sem fios LAN, que permita a nível superior de abstração, mobilidade de um modo transparente. A unidade

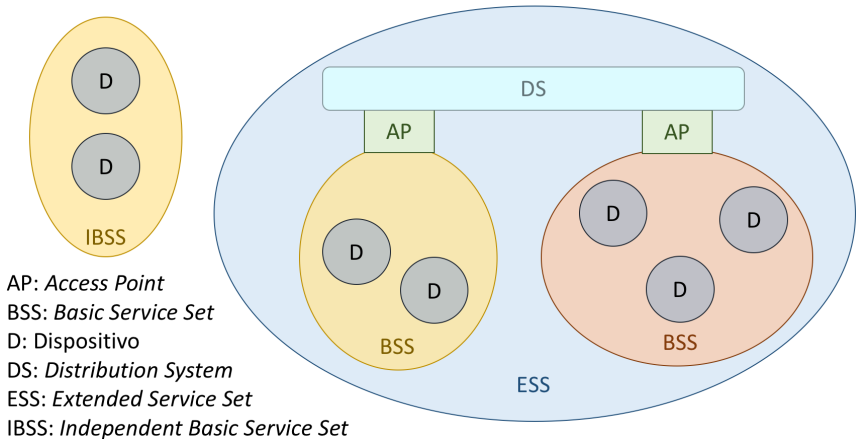


Figura 4.2: Configurações da rede Wi-Fi IBSS e ESS.

básica de uma LAN IEEE 802.11 é chamada BSS (*Basic Service Set*), que é um conjunto de estações móveis ou fixas. Se uma estação sair fora da BSS, impossibilita a comunicação direta com os membros daquela BSS. Baseado na BSS, o IEEE 8002.11 usa as configurações IBSS (*Independent Basic Service Set*) e ESS (*Extended Service Set*). Tal como ilustrado na Figura 4.2, a operação da IBSS é possível quando as estações conseguem comunicar entre si, sem a necessidade de um AP. Isto acontece, pois, este tipo de IEEE 802.11 é normalmente formada sem o planeamento prévio, apenas enquanto for necessário. Este tipo de operação é usualmente denominada de rede *ad hoc*. Em vez de existir de forma independente, a BSS pode também ser uma parte constituinte de uma forma estendida da rede que é constituída por várias BSSs. A componente faz a conexão entre as várias BSSs à DS (*Distribution System*). A DS com AP permite que a IEEE 802.11 para crie uma rede ESS de tamanho e complexidade variável, designando-se por rede estruturada.[12]

O **ZigBee** funciona sobre o IEEE 802.15.4 e define as especificações para a LR-WPAN (*Low-Rate WPAN*). Esta suporta dispositivos simples que consomem pouca energia e tipicamente operam na POS (*Personal Operating Space*) de 10 metros. O ZigBee permite uma rede em *mesh* fiável, que se

auto-organiza e *multi-hop* com grande autonomia energética.[12]

Numa rede ZigBee existem dois tipos de dispositivos que podem participar na rede LR-WPAN: FFD (*Full-Function Device*) e RFD (*Reduced-Function Device*). Os FFDs podem operar em três modos: coordenador da PAN, *router* e *end point*. Um FFD pode falar com RFDs ou outros FFDs, por outro lado os RFD apenas podem falar com um FFD de cada vez. Um RFD é utilizado em aplicações que são extremamente simples tais como, interruptores de luz ou sensores passivos de infravermelhos. Estes não têm necessidade de trocar grandes quantidades de informação. Como consequência, um RFD pode ser implementado usando recursos mínimos e pouca capacidade de memória. Após um FFD ser ativado pela primeira vez pode estabelecer a sua própria rede e tornar-se o coordenador da PAN. Todas as redes em estrela podem operar independentemente de todas as outras que estejam em operação. Isto é conseguido escolhendo um identificador da PAN que não esteja em utilização dentro do raio rádio da rede. Após a escolha do identificador da PAN, os dispositivos podem juntar-se à rede. Um RFD pode-se conectar a uma rede em *cluster-tree*, como um nó de saída no final de um ramo. Qualquer FFD pode servir de *router* e desempenhar um serviço de sincronização aos outros dispositivos ou *routers*. Apenas um *router* pode desempenhar o papel de coordenador da rede PAN, sendo que, este deve possuir um maior número de recursos que os restantes elementos da rede PAN.[12]

Cada protocolo é baseado em normas IEEE. Na Tabela 4.1 estão reunidas as principais diferenças entre os quatro protocolos abordados anteriormente. Tendo em conta os dados na tabela é possível concluir que, os protocolos *UWB* e *Wi-Fi* permitem uma maior velocidade de transferência de dados por outro lado, temos os protocolos Bluetooth e ZigBee que possuem uma velocidade de transferência de dados mais reduzida. De uma maneira geral, podemos afirmar que os protocolos Bluetooth, *UWB* e ZigBee são mais

Tabela 4.1: Comparação dos protocolos Bluetooth, *UWB*, ZigBee e *Wi-Fi*.[12]

Norma	**Bluetooth**	***UWB***	**ZigBee**	***Wi-Fi***
IEEE spec.	802.15.1	802.15.3a *	802.15.4	802.11a/b/g
Banda de Frequência	2.4 GHz	3.1-10.6 GHz	868/915 MHz; 2.4 GHz	2.4 GHz; 5 GHz
Velocidade Máxima de Envio	1 Mb/s	110 Mb/s	250 Kb/s	54 Mb/s
Alcance Nominal	10 m	10 m	10 - 100 m	100 m
Potência Máxima TX	0 - 10 dBm	-41.3 dBm/MHz	(-25) - 0 dBm	15 - 20 dBm
Número de Canais RF	79	(1-15)	1/10; 16	14 (2.4 GHz)
Largura de Banda do Canal	1 MHz	500 MHz - 7.5 GHz	0.3/0.6 MHz; 2 MHz	22 MHz
Tipo de Modulação	GFSK	BPSK, QPSK	BPSK (+ ASK), O-QPSK	BPSK, QPSK COFDM, CCK, M-QAM
Propagação	FHSS	DS-UWB, MB-OFDM	DSSS	DSSS, CCK, OFDM
Mecanismos de Coexistência	Adaptive freq. hopping	Adaptive freq. hopping	Dynamic freq. selection	Dynamic freq. selection, transmit power control (802.11h)
Topologia Básica de Rede	Piconet	Piconet	Star	BSS
Topologia Extendida de Rede	Scatternet	Peer-to-peer	Cluster tree, Mesh	ESS
Numero Máximo de Nós na Rede	8	8	>65000	2007
Encriptação	E0 stream cipher	AES block cipher (CTR, counter mode)	AES block cipher (CTR, counter mode)	RC4 stream cipher (WEP), AES block cipher
Autenticação	Shared secret	CBC-MAC (CCM)	CBC-MAC (ext. do CCM)	WPA2 (802.11i)
Protecção de Dados	16-bit CRC	32-bit CRC	16-bit CRC	32-bit CRC

* Não aprovado.

• Acrónimos: ASK (amplitude shift keying), GFSK (Gaussian frequency SK), BPSK/QPSK (binary/quardrature phase SK), O-QPSK (offset-QPSK), OFDM (orthogonal frequency division multiplexing), COFDM (coded OFDM), MB-OFDM (multiband OFDM), M-QAM (M-ary quadrature amplitude modulation), CCK (complementary code keying), FHSS/DSSS (frequency hopping/direct sequence spread spectrum), BSS/ESS (basic/extended service set), AES (advanced encryption standard), WEP (wired equivalent privacy), WPA (Wi-Fi protected access), CBC-MAC (cipher block chaining message authentication code), CCM (CTR with CBC-MAC), CRC (cyclic redundancy check).

vocacionados para WPAN (cerca de 10m de alcance) e o *Wi-Fi* é mais orientado para WLAN (cerca de 10m de alcance). No entanto, dependendo das aplicações, o ZigBee pode atingir os 100m. [12]

O limite para a potência de emissão para os emissores do protocolo *UWB* que operam na banda *UWB* é -41.3 dBm/MHz, no caso do *Wi-Fi* esse valor é 20 dBm/MHz e em ambos os casos, Bluetooth e ZigBee, a sua potência de envio é 0 dBm/MHz.[12]

Os protocolos Bluetooth e ZigBee são os que possuem um menor consumo energético comparativamente com os protocolos *UWB* e *Wi-Fi*. O Bluetooth e ZigBee são mais vocacionados para aplicações em que não seja necessário a transmissão de uma grande quantidade de dados ou onde haja uma principal preocupação com o consumo energético (como é o caso dos dispositivos móveis que têm a limitação da bateria) uma vez que são protocolos de baixo consumo, permitindo assim um maior tempo de vida das baterias. Por outro lado, quando as implementações requerem grandes quantidades

de informação transferidas os protocolos *UWB* e *Wi-Fi* revelam-se uma melhor solução dado que, embora o seu consumo energético seja superior ao do ZigBee e Bluetooth, apresenta um baixo consumo energético normalizado (mJ/Mb). Dado a topologia de rede, o ZigBee revela-se como a tecnologia mais fácil para se expandir, uma vez que constitui uma rede *mesh*, ao invés das restantes que seguem a topologia de estrela. [12, 14]

4.1.2 ZigBee

Após a comparação das comunicações wireless acima descritas foi tomada a decisão da utilização do protocolo ZigBee na comunicação entre o Módulo de Infusão e o Router. Esta decisão prende-se com o facto de cumprir todas as características que tinham sido definidas nomeadamente o facto de ser de baixo consumo (quando são transmitidas baixas quantidades de informação), tendo também a vantagem de ser uma tecnologia que garante uma enorme facilidade de expansão devido à sua topologia de rede em *mesh*.

Esta norma foi desenvolvida com o objetivo de ser uma solução de baixo consumo, custo e velocidade. O ZigBee atinge velocidades de transmissão na ordem dos 250 kbit/s, não sendo, portanto, destinado a transmissão de uma grande quantidade de dados e o seu alcance andar na ordem dos 140m em espaço aberto.[15]

A tecnologia com as especificações ZigBee tem o intuito de ser mais simples e barata em relação a outros tipos de tecnologias que permitem a criação de uma PAN wireless (WPAN). Esta tecnologia tem várias aplicações, entre elas:

- *Domótica*: segurança, ventilação, controlo de acesso, controlo de iluminação, aquecimento, irrigação de jardim, etc.;
- *Cuidado médico/pessoal*: controlo de pacientes, monitorização corporal;

- *Controlo Industrial*: controlo de processo, gestão de energia, rastreamento de equipamentos;
- *Controlo de dispositivos*: periféricos para PC, rato, teclado, *joystick*;
- *Eletrónica de consumo*: TV, VCR, CD/DVD, controlo remoto, *Home Cinema*, etc.

O facto de um dispositivo ser certificado pela *ZigBee Alliance* assegura que, o dispositivo garanta uma comunicação wireless e interoperabilidade com outros dispositivos ZigBee de confiança. A certificação contribui fortemente para o desenvolvimento das normas. Ao contrário de outras tecnologias, a certificação e os testes são necessários antes do produto poder sair para o mercado.[16]

Rede

Embora no protocolo IEEE 802.15.4 apenas se faça a distinção entre dispositivos FFD e RFD, o protocolo ZigBee faz a distinção entre três tipos de dispositivos lógicos. Nos dispositivos ZigBee o seu papel na rede não é definido por hardware, mas sim por software, dependendo da configuração da rede em que estão inseridos. A nível de hardware todos os dispositivos ZigBee são iguais.

No que toca aos dispositivos contemplados pela norma IEEE 802.15.4 esta, assim como já referido anteriormente, faz a distinção entre dois tipos de unidades lógicas:

- *FFD*: Dispositivos mais complexos da rede. Requer hardware com mais recursos para a implementação da pilha protocolar, tabelas de encaminhamento e configuração de parâmetros da rede. Tem maior consumo energético e capacidade de comunicar com quaisquer outros dispositivos na rede.

- *RFD*: Dispositivos mais simples. Requer um hardware com menos recursos para implementar o protocolo. Só podem comunicar com dispositivos do tipo FFD. Cumprem o papel de extremo da rede, estando muitas vezes associados à leitura de sensores ou outros tipos de informação.

Existem três tipos de dispositivos lógicos que podem participar numa rede ZigBee:

- *Coordenador*(C):
  - Controladores centrais da rede;
  - Apenas pode existir um dispositivo deste tipo por rede;
  - Responsável por inicializar a rede;
  - Faz a sincronização dos relógios dos restantes dispositivos a partir de uma trama de *broadcast* enviada periodicamente;
  - Define um canal na banda de comunicação e um identificador da PAN (PAN ID);
  - Pode funcionar como reencaminhador de mensagens através da rede *mesh*;
  - Responde a pedidos de adesão e dissociação à rede por parte dos *Routers* e *Endpoints*;
  - Não pode entrar em modo *sleep*;
  - Dispositivo FFD.
- *Router*(R):
  - Funciona como intermediário de mensagens ao longo da rede *mesh*;
  - Apenas pode aderir à rede depois de esta ser inicializada pelo Coordenador;

  - Pode admitir outros *Routers* e *Endpoints* na rede;
  - Não pode entrar em modo *sleep*;
  - Dispositivo FFD.

- *Endpoint*(E):
  - Dispositivo de baixo consumo energético;
  - Tem uma aplicação específica identificada por um código (*Cluster ID*);
  - Obtém informação de sensores;
  - Não suporta conexões de outros dispositivos (é um extremo da rede e apenas pode falar com um único dispositivo *Router* ou Coordenador);
  - Pode trocar mensagens com outros elementos da rede;
  - Não tem autoridade para admitir novos dispositivos na rede;
  - Não tem funções de redirecionamento de mensagens pela rede;
  - Pode entrar em modo *sleep* (permite economizar energia) e não recebe qualquer trama, sendo estas retidas no router/coordenador. Ao voltar do modo *sleep* o *endpoint* manda um pedido de *poll* ao coordenador e depois de receber a confirmação do pedido, recebe as tramas pendentes;
  - Dispositivo RFD ou FFD. No caso de ser FFD tem de cumprir as características deste tipo de dispositivos (não podendo entrar em modo *sleep* e tendo a mesma autoridade que um *Router*), apenas diferindo que não tem nenhum dispositivo ligado a ele que não um coordenador ou *Router* e se encontra num extremo da rede a obter informação tal como um *Endpoint* RFD característico.

Cada dispositivo ZigBee possui um endereço de 64-bit e um de 16-bit. O endereço de 64-bit é único para cada dispositivo e não tem qualquer relação

com o outro endereço mais curto, sendo equivalente a um endereço MAC. O endereço curto de 16-bit é atribuído assim que o dispositivo se junta a uma determinada rede, sendo um identificador do dispositivo na rede. [15]

A rede ZigBee pode assumir várias **topologias de rede**, no entanto existem sempre dois elementos que são comuns na rede: um nó coordenador e um *EndPoint*. Para além destes dois elementos pode também ser adicionado à rede o Router dependendo da configuração da rede.

Existem três tipos de topologias (Figura 4.3) que podem ser encontradas numa rede ZigBee[17, 18]:

- *Estrela*: possui apenas um coordenador e um ou mais *EndPoints*. Todos os dispositivos apenas podem comunicar com o coordenador da rede. A rede possui um único nó. Caso um *Endpoint* necessite de comunicar com outro *EndPoints* terá de enviar a informação em primeiro lugar para o coordenador e posteriormente, este irá reencaminhar para o destinatário da informação.

- *Cluster Tree*: possui uma configuração complementar à de estrela. Esta topologia permite que outros dispositivos possam aderir à rede, não só ligando-se ao coordenador, mas também a partir dos *Routers* presentes na rede. Nesta configuração, os routers têm duas funções fundamentais. Em primeiro lugar permite aumentar o número de nós da rede. Por outro lado, permite estender o alcance físico da rede, podendo assim os dispositivos juntarem-se à rede sem ser necessário estarem ao alcance do coordenador. Todas as mensagens são encaminhadas ao longo da árvore de rede até chegarem ao seu destino.

- *Mesh*: a sua estratégia de encaminhamento é um complemento à rede em Cluster dado que cada um dos nós pode comunicar com qualquer outro (dentro do seu espectro), ao contrário da topologia *cluster* em que as mensagens têm de seguir a árvore de rede. Não é permitido

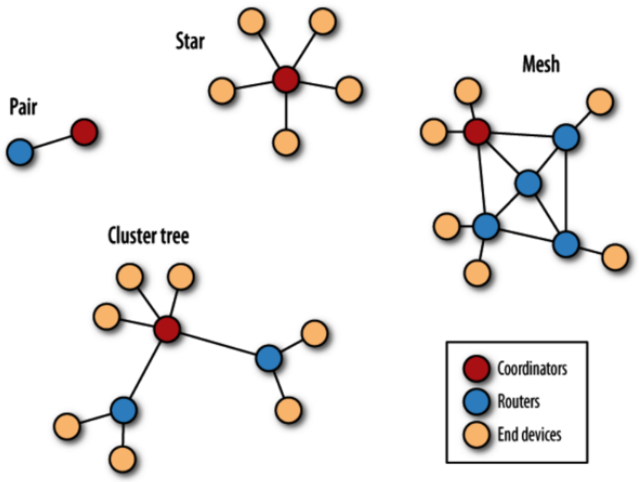


Figura 4.3: Topologias de Rede ZigBee

tramas de *beaconing* devido à complexidade desta rede. As mensagens que têm como destino um *Endpoint* devem ser entregues pelo dispositivo pai, (uma vez que um *Endpoint* apenas se pode ligar a um dispositivo de cada vez). A utilização desta topologia de rede apresenta como vantagens a redução da latência da mensagem e o aumento da fiabilidade e da robustez no que toca a falhas dos nós. Esta configuração de rede procura o caminho mais eficiente e automaticamente encaminha a mensagem pelo mesmo, de modo a evitar falhas.

As topologias de rede Cluster e Mesh são consideradas redes *multi-hop* (devido à sua capacidade de fazer o encaminhamento das mensagens ao longo dos nós até ao seu destino), enquanto que a configuração em estrela é considerada *single-hop*. Uma rede com o protocolo ZigBee é uma rede de multi-acesso, isto significa que todos os nós têm igual acesso ao meio de comunicação.[18]

A topologia ZigBee a ser adotada depende da aplicação a que a rede se destina, não podendo portanto ser considerada a adoção de alguma como errada ou correta, pois cada uma delas tem as suas próprias características.

A **constituição de uma rede** requer um determinado procedimento,

após o qual está pronto para receber pedidos de adesão. Este procedimento pode ser dividido em alguns passos que resumidamente passam por:

1. O coordenador pesquisa dentro da sua banda todos os canais em busca de um que possa utilizar de modo a não interferir com outras frequências wireless, uma vez que as WLAN operam dentro da mesma banda.

2. O coordenador inicia a rede definindo um identificador da rede (PAN ID). A definição deste identificador pode ser tanto pré-configurada (manualmente) como ser atribuído dinamicamente (verifica as PAN ID de outras redes existentes próximas).

3. O coordenador define o seu endereço na rede com o valor 0 (0x0000 em hexadecimal).

Depois destes passos o coordenador emite uma mensagem de *broadcast* para o canal. Com este envio o coordenador pretende saber qual os PAN ID dos *Routers/Endpoints* mais próximos e perceber se algum dispositivo pretende aderir à sua rede. Para que a adesão tenha lugar os *Routers/Endpoints* devem enviar um pedido de adesão ao coordenador.

Ao ligar a uma rede, o dispositivo faz a deteção ativa ou passiva dos canais disponíveis. Os dispositivos que ainda não estejam associados iniciam o processo enviando um pedido de associação ao coordenador da rede. Se este for devidamente recebido o coordenador deverá enviar uma trama de confirmação e a partir deste momento o dispositivo faz parte da rede.

Caso o dispositivo pretenda abandonar a rede, este deverá enviar um pedido de dissociação de rede para o coordenador.

Stack

O protocolo ZigBee é uma extensão do seu *standard*, onde para além das suas camadas (PHY e MAC) da norma define também as camadas de rede,

aplicação e serviços protocolares de segurança.

A camada de rede do ZigBee suporta características avançadas, nomeadamente a gestão de energia, endereçamento, correção de erros, formatação de mensagens entre outras especificações de extrema importância para que haja uma comunicação fiável entre dois dispositivos.[14] A camada de rede é responsável por formar a topologia de rede, configuração do dispositivo, descobrir nós próximos e caminhos entre eles, entregando as mensagens no destino correto, endereçar, encaminhar e receber informação das camadas de aplicação e MAC. [15]

4.1.3 Comunicação Cliente-Servidor

A arquitetura cliente-servidor é uma arquitetura de rede na qual cada máquina ou processo na rede pode ser tanto um cliente como servidor. Os servidores caracterizam-se por serem máquinas com bastantes recursos físicos ou processos dedicados a gerir discos de armazenamento, impressoras, ou tráfego da rede. Os clientes caracterizam-se por ser máquinas onde os utilizadores correm aplicações que dependem da comunicação com o servidor para as suas tarefas.

O estabelecimento de comunicação entre o cliente e o servidor pode ser feito a partir de várias maneiras. Atualmente, e para a aplicação específica que se pretende fazer, pode-se afirmar que os métodos mais usados são através de:

- Middleware
- HTTP
- Websocket

O termo **Middleware** é normalmente aplicado ao software que se destina a interligar programas geralmente complexos e já existentes. Frequentemente é utilizado para ligar aplicações empresariais e WebServices.

O middleware assenta entre o sistema operativo e as aplicações em diferentes servidores e simplifica o desenvolvimento de aplicações que requerem serviços de outras aplicações. Isto permite que os programadores criem aplicações de negócio sem ser necessário desenvolver de raiz a integração por cada nova aplicação. Tipicamente o middleware garante serviço de mensagens para que as aplicações possam comunicar através de frameworks de mensagens tal como SOAP (*Simple Object Access Protocol*), serviços web, REST (*Representational State Transfer*), JSON (*JavaScript Object Notation*), XMPP (*Extensible Messaging and Presence Protocol*), AMQP (*Advanced Message Queuing Protocol*), MQTT (*Message Queuing Telemetry Transport*), etc.

O protocolo **HTTP** (*Hypertext Transfer Protocol*) é um dos protocolos mais populares, utilizados mundialmente e tem sido utilizado pela iniciativa *World-Wide Web* desde 1990. O HTTP é um protocolo do nível de aplicação, que se caracteriza por ser genérico e que pode ser usado para várias tarefas para além do *hipertext*, tal como nome de servidores e sistemas de gestão para sistemas distribuídos através de uma extensão dos métodos de pedidos, códigos de erro e headers. [19]

O **WebSocket** permite a comunicação bidirecional entre o cliente e o servidor remoto. O protocolo consiste num *handshake* de abertura seguido de uma trama básica de mensagem, assente na camada TCP. O objetivo desta tecnologia é permitir um mecanismo para aplicações baseadas em browser, que necessitem de uma comunicação nos dois sentidos com o servidor que não aguenta com múltiplas conexões HTTP (por exemplo usando *XMLHttpRequest* ou *iframes* e *pollings* de longa duração). [20]

O protocolo WebSocket foi desenvolvido com o intuito de substituir as atuais tecnologias de comunicação que usam HTTP como camada de transporte para tirar proveito da infraestrutura existente (*proxies*, autenticação, filtragem, etc.). Tais tecnologias foram implementadas como balanceamento

entre eficiência e fiabilidade uma vez que o HTTP não foi inicialmente desenvolvido para comunicações bidirecionais. O WebSocket tenta abordar os objetivos das tecnologias bidirecionais dentro do contexto da infraestrutura HTTP. Como tal, foi desenhada para funcionar sobre o HTTP nas portas 80 e 443 assim como suportar proxies HTTP e intermediários, mesmo que isso implique alguma complexidade específica no ambiente em que está inserido. Contudo, a sua arquitetura não limita o WebSocket ao HTTP, e futuras implementações poderão simplificar o *handshake* sobre uma porta dedicada sem para isso ter de reinventar todo o protocolo. Este último ponto é importante na medida em que os parâmetros de tráfego de mensagens interativas não coincidem inteiramente com o tráfego característico no HTTP e pode induzir cargas anormais nos seus componentes. [20]

De modo a poder fazer a **seleção do protocolo de comunicação** a ser adotado é necessário ter em consideração as necessidades/características do mesmo. Uma das características deste sistema será a direção das comunicações. Neste projeto o servidor é um nó central e todos os dispositivos presentes na rede apenas devem fazer pedidos ao servidor sendo, portanto, os clientes a tomar a iniciativa de estabelecer ligação com o servidor (e não com qualquer outro cliente), pode-se por isso considerar que a comunicação é unidirecional e que não existe a necessidade de ter implementado nem configurado nenhum serviço de troca de mensagens. Outro fator que foi tido em conta para a seleção da tecnologia foi a simplicidade de utilização e configuração de todo o ambiente necessário para a aplicação do protocolo.

Após ter consciência das características das tecnologias estudadas foi decidido utilizar o protocolo HTTP. Este foi adotado tendo em conta que a comunicação unidirecional é suficiente para cumprir os requisitos do sistema, não é necessário um grande esforço para configurar o servidor com serviços de mensagens e para além disso, a experiência anteriormente adquirida no uso deste protocolo permitiu uma maior agilidade e compreensão do

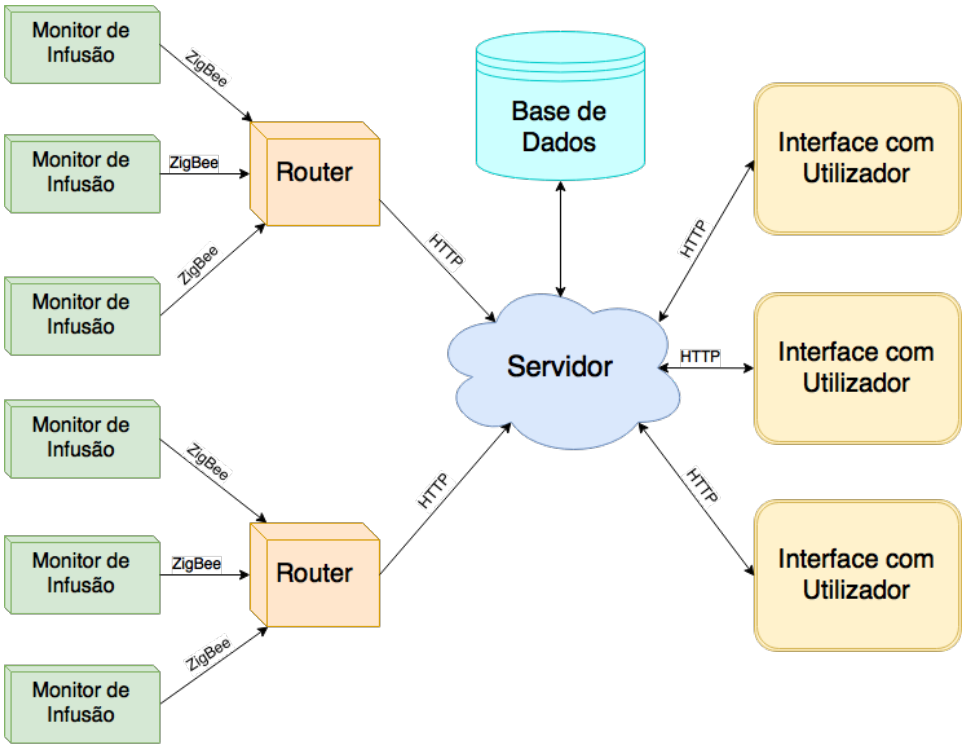


Figura 4.4: Diagrama Geral de Comunicação do Sistema

comportamento do mesmo.

Como resultado das opções da tecnologia de comunicação, o Diagrama Geral do Sistema (Figura 3.1 na página 23) apresentado no Capitulo 3 passa a apresentar o esquema da Figura 4.4.

4.2 Hardware

De modo a ser possível o desenvolvimento do projeto foi necessário haver uma pré-seleção do hardware a ser utilizado. Nesta secção pretende-se explorar algumas escolhas possíveis no mercado e as respetivas decisões dos elementos que foram aplicados ao projeto.

4.2.1 Módulo ZigBee

Um dos elementos de hardware que é necessário escolher é o módulo de comunicação que vai implementar o protocolo ZigBee. No que toca a este tipo de módulos estes podem-se classificar de duas maneiras: sistema completo ou transcetor Designa-se por sistema completo um módulo que inclui um microcontrolador, transcetor e antena implementados num PCB. Nos módulos em que apenas se encontra presente o transcetor é necessário processar toda a *stack* do protocolo externamente ao módulo, o que implica um grande esforço adicional. Por outro lado, este tipo de módulos, caracteriza-se por ser mais económico que o sistema completo.

Os sistemas completos são módulos que já vêm com um microcontrolador integrado (sendo que estes podem ou não ser reprogramados) e já têm embebidos em si todo o programa capaz de lidar com a camada protocolar. Estes têm a vantagem de serem controlados mais ao alto nível que o tipo anterior, sendo desta forma possível a criação de uma rede de maior dimensão, mais complexa e com a confiança que a implementação do protocolo será provavelmente mais isenta de erros do que caso fosse necessário implementar toda a pilha protocolar. No entanto, como este tipo de módulos apresenta um microcontrolador integrado, o seu preço torna-se superior.

Uma vez que o que se pretende no decorrer deste projeto é o desenvolvimento de um sistema de monitorização de infusão e não engloba fazer o estudo aprofundado do protocolo ZigBee, foi decidido utilizar um módulo que contivesse um sistema completo. Desta forma garantiu-se que o desenvolvimento tira-se proveito de todas as vantagens de uma rede ZigBee sem sacrificar tempo precioso para outras tarefas mais críticas e sobre o as quais é mais importante debruçar.

No que toca a estes tipos de módulos a oferta no mercado é muito diversificada, no entanto optou-se pelo uso do módulo XBee por ter bastante credibilidade no mercado, ser um dispositivo que apresenta uma documentação

bem executada e permite também fazer uma implementação a alto nível.

4.2.2 XBee

O módulo XBee é muito popular e é baseado no protocolo ZigBee, certificado pela ZigBee Alliance e produzido pela *Digi Internacional* que é uma das marcas lideres de mercado no segmento. Estes módulos têm na sua constituição um microcontrolador e um transcetor. O seu *firmware* possui a implementação do protocolo ZigBee e permite a enumeração do comportamento do dispositivo (Coordenador, Router e Endpoint).

Cada dispositivo é único na rede e possui dois endereços: o endereço de rede e o número de série. O endereço de rede é atribuído pelo coordenador de forma automática assim que o módulo se liga à rede. Por outro lado o número de série é único para cada dispositivo produzido.

Seleção do Modelo

Quando se fala em XBee não se fala apenas num dispositivo em específico mas sim de toda uma gama de modelos (Figura 4.5) sobre o qual assenta a nomenclatura XBee, sendo portanto necessário efetuar uma seleção do modelo.

Assim como é possível verificar na Figura 4.5, os módulos XBee existem sob diversificados tipos sendo que algumas das especificações que variam e que podem ser destacadas são o encapsulamento, tipo de antena, protocolo utilizado, alcance, consumo energético e frequência de operação.

No que toca ao encapsulamento os módulos podem ser classificados como Discretos ou SMD. O encapsulamento Discreto possui um *socket* com 20 pinos e necessita de furos para a montagem em PCB ou de um *socket* onde possa ser encaixado e montado no PCB. O encapsulamento SMD não requer furos nem *sockets*, sendo soldados diretamente no PCB. Geralmente os encapsulamentos possuem fins diferentes dependendo da fase de produção

Family	Frequency	Protocol	Description	RF Line of Sight Range	Form Factor	Development Kit Part Numbers	RF Data Rate	Current Draw Tx/Rx	Hardware Reference # / Chipset(s)	Certified Regions
XBee® Cellular	Bands 4 and 13	LTE Cat 1	Pre-certified for LTE Cat 1, in an XBee form factor	Cellular Network Coverage	Through-Hole	XKC-V1T-U	10 Mbps Down / 5 Mbps Up	860mA / 530mA	SiLabs EFM32GG395F1024 ARM M3 MCU	US
XBee® Wi-Fi	2.4 GHz	IEEE 802.11	Wi-Fi 802.11b/g/n with easy provisioning and point-to-multipoint device connectivity	N/A	Through-Hole	XKA2B-WFT-0	1 to 72 Mbps	309 mA / 100 mA	**S6B** SiLabs EFM32LG230 ARM M3 MCU, Atheros AR4100 Transceiver	US, CA, EU, AU, JP
XBee® DigiMesh® 2.4		DigiMesh®	DigiMesh networking, low-cost, low-power	4000 ft (1200 m)		XK-WDM	250 Kbps	33mA / 28mA	**S2C** SiLabs EM357 SoC	US, CA, EU, AUS/ NZ, BR, JP
XBee-PRO DigiMesh® 2.4			Extended-range DigiMesh	2 miles (3200 m)				120 mA / 31 mA		US, CA, AU, NZ, BR
XBee® 802.15.4		Proprietary 802.15.4	low cost, low power point-to-multipoint device connectivity	4000 ft (1.2 km)	Surface Mount	XKB2-A2T-WWC		33mA / 28mA		US, CA, EU, AUS/ NZ, BR, JP
XBee-PRO® 802.15.4			Point to multipoint extended range version	2 miles (3.2 km)				120mA / 31 mA		US, CA, AU, NZ, BR
XBee® ZigBee		ZigBee® Pro	ZigBee mesh networking, low-cost, low-power	4000 ft / 1.2 km		XKB2-Z7T-WZM		33mA / 28mA		US, CA, EU, AUS/ NZ, BR, JP
XBee-PRO® ZigBee			Extended-range ZigBee	2 miles / 3.2 km		XKA2C-Z7T-U		120 mA / 31 mA		US, CA, AU, NZ, BR
XBee® ZigBee - Thread Ready		ZigBee® Pro Thread	ZigBee protocol (upgradable to Thread protocol) low cost, low power	4000 ft (1.2km)	Surface Mount	XKB2-Z7T-WTZM		33mA / 28mA	**S2D** SiLabs EM3587 SoC	US, CA, EU
XBee-PRO® 900HP	900 MHz	Multipoint DigiMesh®	Extended-range peer-to-peer mesh, sleeping routers	9 miles / 14.5 km	Through-Hole	XKB9-DMT-UHP (US/CA) XKB9-DMT-AHP (AU) XKB9-DMT-BHP (BR) XKB9-DMT-SHP (SGP)	10 Kbps or 200 Kbps	215 mA / 29 mA	**S3B** SiLabs EFM32G230F128 ARM M3 MCU, Analog Devices ADF7023 Transceiver	US, CA, AU, BR
XBee® SX			20mW networking XBee module for mission critical applications	9 miles / 14 km	Surface Mount	XK9X-DMS-0	250 Kbps	55 mA / 40 mA	**S1D** SiLabs EFM32LG230F256G ARM M3 MCU, Analog Devices ADF7023 Transceiver, LNA/SAW (PRO version: PA+LNA/SAW)	US, CA, AU, NZ (BR Pending)
XBee-PRO® SX			1-Watt networking XBee module for mission critical applications	65 miles / 105 km				900 mA / 40 mA		US, CA, AU, (BR Pending)
XBee® 868LP	868 MHz	Multipoint DigiMesh®	Low-cost, low-power peer-to-peer mesh for Europe	5.2 miles / 8.4 km	Surface Mount	XK8-DMS-0 XK8-DMSB0	10 Kbps or 80 Kbps	48 mA / 27 mA	**S8** SiLabs EFM32G230F128 ARM M3 MCU, Analog Devices ADF7023 Transceiver	EU

Figura 4.5: Tabela comparativa de modelos XBee.[5]

em que se encontra. O encapsulamento discreto é normalmente utilizado numa etapa mais inicial da produção, nomeadamente a prototipagem, enquanto que por outro lado o encapsulamento SMD já mais recomendado para aplicações de elevado volume uma vez que em custos é mais reduzido, não necessita de *sockets* e podendo ser instalados a partir do uso de uma *pick-and-place*.

Quanto à antena os módulos XBee podem oferecer cinco tipos de antenas diferentes mais concretamente a antena em PCB, U.FL, *Whip*, fio de antena integrado e RP_SMA. No caso da antena em PCB a antena está embutida no próprio PCB com pistas condutoras. Na antena U.FL existe um pequeno conector para uma antena externa, sendo é aconselhada em soluções que estejam integradas dentro de uma caixa, mas que se pretenda instalar uma antena externa. A antena *Whip* consiste numa antena sólida com cerca de 25mm de altura e pode ser modelada para maximizar a força do sinal ou permitir colocar exteriormente à sua caixa e permitir assim um maior

alcance comparativamente com antenas integradas ,podendo também ser mais facilmente direcionadas. Na antena Fio de Antena Integrado pode-se encontrar um pequeno fio de 80mm perpendicular ao módulo, utilizando para tal um fio soldado de $\frac{1}{4}$-onda diretamente ao PCB do módulo. Por fim a antena RP_SMA é o maior conector de antena que permite adaptar uma antena externa que permite que esta fique fora da caixa em que o módulo está instalado e obter um bom alcance. [21]

Os módulos XBee são fabricados em duas versões: a versão XBee PRO e a XBee. A diferença significativa entre estas duas versões reside na potência de transmissão que é bastante superior na versão PRO e consequentemente faz elevar o seu consumo energético em relação à outra versão.

Os modelos XBee oferecem soluções com diferentes frequências, podendo funcionar a 2,4GHz, 915MHz ou 868MHz. A frequência tem influência direta nas características de transmissão pois quanto maior for a banda de transmissão maior será a taxa de transmissão, no entanto menor será o alcance. Apesar de existirem várias frequências nas quais os XBee podem ser adquiridos, existem normas (dependendo da região em que se pretende instalar o sistema) de gamas de emissão de radiofrequência que devem ser cumpridas (na Europa é 2,4GHz).

Uma vez analisados todos os parâmetros/características existentes e necessárias para o desenvolvimento do sistema foi escolhido o XBee com encapsulamento discreto (dada a fase de prototipagem em que se encontra), com antena *whip* (uma vez que não será necessária, nesta fase, uma antena com mais alcance e é um dos modelos mais comerciais) e na versão XBee (que possui uma menor potência de transmissão no entanto possui um menor consumo energético que poderá ser uma das preocupações quando se pretende permitir mobilidade do monitor de infusão).

Figura 4.6: Modo de operação transparente.[6]

Modo de Operação

Os módulos XBee permitem a seleção do Modo de Operação que vai permitir o modo de comunicação dos módulos sendo eles: modo transparente (AT) e modo API (*Application Programming Interface*). Estes modos devem ser selecionados aquando da configuração dos módulos.

No que toca ao **modo transparente** a interação entre o XBee e o microprocessador/computador é feita de forma direta semelhante ao funcionamento de uma porta série sem nenhuma informação adicional, proporcionando uma maior camada de abstração para o utilizador (Figura 4.6). Este modo proporciona o método de comunicação mais simples entre os módulos uma vez que a informação não sofre alteração entre a emissão e a receção. Por outro lado, este método não oferece um nível de robustez alto, sendo desta forma um método pouco eficaz de estabelecer um meio de comunicação fiável entre dois computadores. Este modo surge mais com a intenção de estabelecer a comunicação entre apenas dois módulos e como tal quando se pretende alterar o destino da mensagem deve-se proceder à reconfiguração do XBee para alterar o seu destino. Uma vez que se destina a uma comunicação ponto-a-ponto e que não apresenta mais informação adicional não é possível detetar a origem da mensagem, sendo que, caso se pretenda descobrir esta, é necessário que o utilizador implemente um protocolo que lhe permita ter acesso a este tipo de informação e todos os módulos devem respeitá-lo. [6]

O **modo API** oferece uma interface estruturada de dados. A informação é enviada pela porta série a partir de pacotes ordenados, permitindo desta

Figura 4.7: Modo de operação API.[6]

forma que seja necessária a criação de comunicações complexas entre dois módulos sem haver a necessidade de implementar nenhum protocolo próprio (Figura 4.7). Este modo apresenta algumas vantagens de utilização em comparação em relação ao modo transparente sendo algumas delas:[6]

- Uma vez que existem diferentes tipos de tramas com diferentes objetivos (como comunicação e configuração), é possível configurar os módulos XBee diretamente sem ser necessário entrar em modo comando;
- Dado que o IP é um dos parâmetros da trama API é possível comunicar facilmente com múltiplos módulos;
- A trama contém o endereço de origem, sendo por isso facilmente identificável a proveniência da mensagem;
- É possível configurar localmente ou remotamente os módulos da rede;
- É simples de gerir comunicações entre um ou múltiplos dispositivos;
- Existem mensagens de confirmação/erro de entrega de mensagem;
- A força do sinal do pacote recebido é possível ser consultada;
- É possível realizar um diagnóstico da rede, assim como fazer a gestão do mesmo.

Os pacotes que circulam no modo API são enviados e recebidos a partir da interface do XBee e possui a mensagem *wireless* assim como alguma

Start delimiter	Length		Frame data								Checksum
			Frame type	Data							
1	2	3	4	5	6	7	8	9	...	n	n+1
0x7E	MSB	LSB	API frame type	Frame-type-specific data							Single byte

Note MSB represents the most significant byte, and LSB represents the least significant byte.

Figura 4.8: Estrutura da trama API.[6]

informação extra tais como origem/destino da informação e a qualidade do sinal.[6]

Quando um XBee está em modo de funcionamento API, toda a informação que entra ou sai do módulo pela porta série está encapsulada em tramas que definem o modo de operação ou eventos do módulo. A trama API segue a estrutura presente na Figura 4.8. Qualquer informação que chegue à porta série é descartada até ser encontrado o *start delimiter*, a informação não chegue corretamente ou falhe a verificação do *checksum*. [6]

Assim como é possível verificar na Figura 4.8 a trama pode ser separada em diferentes elementos, nomeadamente:[6]

- Delimitador Inicial(*Start Delimiter*): primeiro byte da trama que consiste numa sequência especial de bits que indica o inicio da trama. O valor em hexadecimal é 0x7E. Este byte permite facilmente detetar o inicio de uma trama.

- Tamanho(*Length*): este campo indica qual o número total de bytes que estão incluídos no campo dos dados. Este é constituído por dois bytes que não incluem na sua contabilização o delimitador inicial, o campo de tamanho e o *checksum*.

- Informação (*Frame Data*): este campo contém a informação recebida ou a ser transmitida. A informação é estruturada com base no objetivo

da trama API, sendo que esta pode ser dividida em duas partes:

- Tipo de trama (*Frame Type*): este parâmetro contém o identificador de trama (Figura 4.9). Este campo vai determinar qual é o tipo de trama API e indica como a informação é organizada no campo dados.
- Dados (*Data*): contém a informação da trama. A informação incluída neste campo e a sua ordem vai depender do Tipo de Trama.

- *Checksum*: é o último byte da trama e tem o intuito de verificar a integridade da informação recebida. Esta verificação é feita codificando a soma de todos os bytes presentes no campo Informação.

  O **cálculo** começa com a soma de todos os bytes da informação. De seguida retira-se os 8 bits mais significativos da soma e com eles faz-se a subtração com 0xFF. Do resultado desta operação obtém-se o *checksum*.

  Para efetuar a **verificação** do *checksum* de uma trama recebida soma-se todos os bytes de informação mais o *checksum* recebido na trama. Caso os 8 bits mais significativos da soma seja 0xFF pode-se afirmar que o *checksum* é válido, caso contrário é inferido que houve um erro na transmissão.

Uma vez analisados todos os dados referentes ao modo de operação, possíveis no módulo XBee, foi decidido utilizar o modo API uma vez que permite efetuar a comunicação entre dispositivos com maior flexibilidade e maior conhecimento do que se passa na rede (conhecimento da origem/destino da mensagem).

API Frame Names	API ID
AT Command	0x08
AT Command - Queue Parameter Value	0x09
ZigBee Transmit Request	0x10
Explicit Addressing ZigBee Command Frame	0x11
Remote Command Request	0x17
Create Source Route	0x21
AT Command Response	0x88
Modem Status	0x8A
ZigBee Transmit Status	0x8B
ZigBee Receive Packet (AO=0)	0x90
ZigBee Explicit Rx Indicator (AO=1)	0x91
ZigBee I/O Data Sample Rx Indicator	0x92
XBee Sensor Read Indicator (AO=0)	0x94
Node Identification Indicator (AO=0)	0x95
Remote Command Response	0x97
Extended Modem Status	0x98
Over-the-Air Firmware Update Status	0xA0
Route Record Indicator	0xA1
Many-to-One Route Request Indicator	0xA3

Figura 4.9: Identificadores de tramas API.[7]

4.2.3 Microcontrolador

Um microcontrolador (MCU) é um circuito integrado que realiza funções de cálculo e tomadas de decisão. Este circuito contém um núcleo de processador, periféricos de entrada/saída programáveis e memória. De uma maneira geral pode ser entendido como um microprocessador que é programado para cumprir uma tarefa em específico, isto é, ser um sistema embebido.

Atualmente existe uma vasta oferta de microcontroladores e a maneira de filtrar esta diversidade recai sobre vários aspetos entre eles: número de bits, frequência de relógio, número de pinos, periféricos, quantidade de memória de programa, quantidade de memória volátil, linguagem de programação, software de desenvolvimento, aplicação, custo, requisitos de tempo real, consumos, arquitetura, etc. Um aspeto que pode ter também um peso bastante relevante na escolha do MCU é a experiência de trabalho que se tem com uma determinada marca/gama de microcontroladores pois reduz a curva de aprendizagem e aumenta o nível de especialização o que se traduz na otimização de código e redução do tempo de trabalho necessário para cumprir uma determinada tarefa.

A **arquitetura ARM** é uma arquitetura de processador de 32 bits utilizada geralmente em sistemas embebidos. A organização ARM faz o licenciamento do core ARM e os diversos fabricantes colocam a memória e os periféricos que pretendem (cumprindo a especificação). Desta forma os microcontroladores dos diferentes fabricantes podem ter diferentes tamanhos de memória, número de I/O, temporizadores, etc.

Este tipo de arquitetura visa a simplificação de instruções com o objetivo de obter mais eficiência por ciclo de relógio podendo desta forma realizar tarefas menores com ciclos mais curtos e garantir uma maior organização das operações dentro do núcleo de processamento, obtendo assim o melhor desempenho possível para além de ser simples e ter um baixo consumo energético. O padrão RISC do processador possibilita que estes processa-

dores necessitem de menos transístores que processadores CISC (x86). Esta abordagem reduz custos, libertação de calor e consumo de energia.

No que toca ao monitor de infusão, este é um sistema simples em recursos necessários. Para tal, decidiu-se utilizar um microprocessador ARM de baixo consumo energético. O microcontrolador escolhido foi o STM32L100RC que pertence à categoria de ultra-low-power do fabricante *ST Microeletronics* que garante uma razão otimizada entre custo/desempenho para todos os tipos de aplicações de baixo consumo. Esta gama de dispositivos possui o mais baixo consumo do mercado no modo *Stop* (com retenção da SRAM) de 350nA, enquanto permite o tempo de transição para operação normal de 3,5μs.[22] Este microcontrolador em específico possui a vantagem de estar presente numa placa de desenvolvimento, o que permite mais facilmente o desenvolvimento do protótipo do sistema.

4.2.4 Deteção de Fluxo

A deteção de fluxo (assim como já referido no Capitulo Estado da Arte) pode ser feito a partir de um controlo Volumétrico ou Não-Volumétrico. Cada um dos métodos possui as suas vantagens e requerem metodologias diferentes para efetuar o cálculo da infusão que está a ser administrada ao paciente.

Para efetuar o controlo *Volumétrico* é necessário saber a velocidade de infusão do líquido em ml/h, o que pode ser vantajoso porque não depende das características do líquido. Por outro lado, fazer a medição por este método pode ser considerado um grande desafio pois, se para outros tipos de aplicação (como por exemplo os caudalímetros) se pode colocar um mecanismo no meio do sistema que interfere com os líquidos e os seus caudais são consideráveis, neste caso tem de se considerar que o risco de contaminação do medicamento infundido e o reduzido caudal presente no sistema. Este tipo de requisitos impossibilita a deteção pela via dos métodos convencionais noutro tipo de aplicações, entre esses métodos encontram-se, por exemplo:

- Moinho Eletrónico - a parte mecânica funciona da mesma maneira que um moinho (com pás transversais ao fluxo do líquido) e à medida que o fluxo vai passando empurra as pás, o mecanismo roda. A parte rotativa do mecanismo possui ímanes que ao rodar passam perto de um senso por efeito Hall e por cada impulso o controlador sabe quanto fluxo atravessou o mecanismo no espaço de tempo compreendido entre dois impulsos consecutivos.

- Turbina Eletrónica - uma turbina é colocada em frente ao fluxo encapsulada no tubo. O fluxo ao passar pela turbina fá-la rodar, e a velocidade de rotação será proporcional ao fluxo que a atravessa.

Outros métodos válidos usados no controlo volumétrico são os mesmos utilizados pelas bombas de infusão (Secção 2.1.3) nomeadamente por mecanismo peristáltico, seringa ou pistão. Estes métodos (à exceção do pistão) não interferem com o líquido administrado. No entanto implicam que se faça o controlo físico da regulação do fluxo, o que não é o pretendido neste projeto.

Por sua vez o controlo *Não-Volumétrico*, apesar de poder ser influenciado por algumas características do líquido infundido, consegue garantir de igual forma uma boa precisão e apresenta-se como sendo uma solução cuja deteção pode ser mais fácil que a de controlo volumétrico. Outra característica que é necessária considerar para poder aplicar este tipo de controlo é o volume da gota que dependendo do sistema utilizado pode ser 10, 15 ou 20 gotas/ml para macrogotas e 60 gotas/ml para microgotas. Este tipo de controlo apresenta como unidade de medida gotas/minuto sendo, portanto, necessário saber o número de gotas que caem a cada minuto e o seu volume. Esta medição tem de ser feita na câmara de gotejamento do sistema (Figura 2.3).

Dado que este é o método de controlo que permite efetuar a monitorização do fluxo administrado com maior facilidade que o controlo Vo-

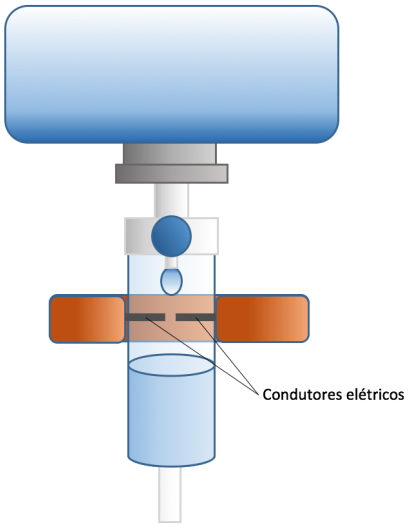


Figura 4.10: Deteção por Continuidade de Corrente Elétrica

lumétrico, será este o método utilizado. Assim, é necessário definir dentro dos métodos de deteção da gota qual deles será o método mais adequado.

Um método possível é a deteção da gota a partir da *Continuidade de Corrente Elétrica* (Figura 4.10). Este método baseia-se no princípio de continuidade da corrente elétrica no líquido. De maneira a aplicar esta metodologia, dentro da câmara de gotejamento teriam de existir dois contactos elétricos separados por uma distância inferior ao diâmetro de uma gota. A gota ao cair iria fechar o circuito e provocar um pulso elétrico que iria indicar ao controlador a passagem de uma gota. Este método apresenta alguns inconvenientes: depende da condutividade do líquido, interfere com a medicação podendo contaminar a mesma e uma vez que a corrente elétrica pode atravessar o líquido pode alterar as características deste.

Outro método possível é por *efeito de Hall* (Figura 4.11) que se caracteriza pelo surgimento de uma diferença de potencial num condutor elétrico, transversal ao fluxo de corrente e um campo magnético perpendicular à corrente. Para a aplicação deste efeito na deteção das gotas seria necessário colocar um íman num dos lados da câmara de gotejamento e de um sensor de

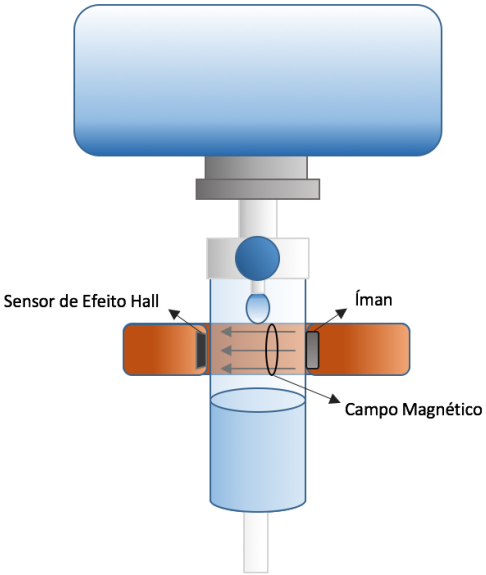


Figura 4.11: Deteção por Efeito Hall

efeito Hall no lado oposto. Para que este método seja aplicável pressupõe-se que a gota ao cair iria interferir no campo magnético o suficiente para que no sensor de efeito Hall houvesse uma alteração temporária na sua tensão de saída e desta forma fosse possível a deteção da gota. Este possível método de deteção apresenta a vantagem de não ser intrusivo no sistema, no entanto pode haver falsas deteções de gotas quando houver objetos metálicos que se aproximem demasiado do sensor e existe também o inconveniente de a maioria dos sensores deste género serem utilizados para outros tipos de aplicação que implicam a deteção de campo magnético a curta distância, o que para este sistema em concreto pode ser um obstáculo pois a distância entre as duas paredes da câmara de gotejamento é superior à que a maioria dos sensores é capaz de detetar. De modo a contornar a distância, seria necessário usar um íman mais forte que o normal (como por exemplo de neodímio de dimensão razoável).

É possível também fazer a deteção a partir de *interrupção do feixe de luz*. A aplicação deste método é em todo semelhante ao método de efeito

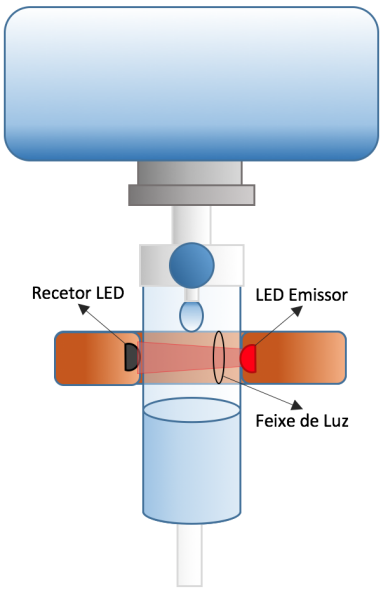


Figura 4.12: Deteção por Interrupção do Feixe de Luz

de Hall, apenas tendo de substituir o íman pelo LED emissor e o sensor de Hall por um recetor de LED. Neste sistema a gota ao cair interrompe o feixe de luz emitido pelo emissor, o que resulta numa alteração da tensão à saída do recetor de LED. A metodologia aqui descrita é aplicada em alguns controladores de infusão e em bombas de infusão para fazer a monitorização da dosagem de medicamentação. Este método requer que tanto o LED emissor como o recetor estejam alinhados de maneira a garantir que o recetor está dentro da divergência do raio do emissor e assim tirar proveito da potência de emissão deste. Caso o emissor não esteja devidamente alinhado com o recetor ou a potência do emissor não seja suficiente a gota não é detetada e as diferenças de luz ambiente podem induzir falsas gotas ou mesmo impedir que a gota seja detetada.

Para este projeto decidiu-se adotar a metodologia de deteção de medicação por interrupção do feixe de luz, pois é um sistema que comprovadamente funciona (uma vez que existem várias soluções de mercado que fazem uso da mesma), os componentes necessários para a sua execução são

de baixo custo e é de fácil implementação.

4.3 Software

Nesta secção pretende-se abordar alguns pontos relativos ao desenvolvimento do Software necessário para o funcionamento do sistema. Dentro destes pontos encontram-se as linguagens de programação, as ferramentas e os serviços utilizados.

O sistema que se pretende desenvolver é algo com uma arquitetura simples de perceber, mas que, no entanto, possui vários elementos constituintes, o que implica que seja necessário o desenvolvimento de, não só um software, mas sim quatro, nomeadamente para: o Monitor de Infusão, Router, Interface com Utilizador e um *Web Service* para o servidor.

4.3.1 Linguagem de Programação

As máquinas são capazes de fazer tarefas diversificadas, trocar facilmente informação independentemente do meio físico, simplificar tarefas diárias assim como automatizar inúmeras tarefas que de outra maneira seriam bastante aborrecidas e que requeriam um esforço muito elevado. No entanto as máquinas não possuem inteligência própria tendo, desta forma, de serem ensinadas de maneira inequívoca a desempenhar uma determinada tarefa. De modo a poder ser transmitido o conhecimento e procedimento para a máquina, é necessário utilizar a sua linguagem nativa que em nada se assemelha à linguagem humana. [23]

Para comunicar exatamente aos nossos computadores o que queremos que façam, desenvolvemos uma ampla gama de linguagens de programação para facilitar o processo de comunicação. Para a escolha existem inúmeras linguagens disponíveis, sendo que cada uma delas tem as suas características (compilada/interpretada, alto/baixo nível, regras de sintaxe, paradigmas

suportados, normalização, ...).[23] No entanto aqui não se pretende fazer um estudo entre as diferentes linguagens e os seu propósitos, mas sim mostrar quais as opções tomadas para cada um dos elementos do sistema.

Para o Monitor de Infusão pretendeu-se utilizar um microcontrolador a nível do hardware e de modo a poder programá-lo foi utilizada a linguagem C. Esta linguagem foi adotada como desenvolvimento de sistemas pois o seu produto atinge velocidades de execução muito semelhantes ao assembly.

Relativamente ao software desenvolvido para o Interface com o utilizador e Router do sistema, optou-se por utilizar a linguagem C++. Esta tem como base o C no entanto tem adicionado em si o paradigma de programação orientada a objetos. O facto de ter a linguagem C como base com orientação a objetos tornou esta linguagem de grande respeito pela sua portabilidade de código (graças ao paradigma orientado por objectos) sem, no entanto, sacrificar a velocidade e funcionalidade de baixo nível característico do C.[24]

Quanto ao software do *Webservice* localizado no Servidor a linguagem adotada foi o PHP que é tida como uma linguagem de *scripts* de uso geral que é utilizada para desenvolvimento web. [25]

4.3.2 Ferramentas Utilizadas

De modo a desenvolver o software foi feito uso de diferentes ferramentas como IDE, gerador de código de inicialização e serviços. Ao longo do projeto foram utilizadas mais do que uma ferramenta dependendo do software em questão.

Para o software desenvolvido para o Monitor de Infusão foi utilizado como **IDE** o *Keil MDK* que se caracteriza por ser um ambiente de desenvolvimento de software para uma vasta gama de dispositivos baseados na arquitetura ARM Cortex-M. Este IDE possui também *debugger*, compilador de C/C++ para ARM, assim como outros componentes essenciais de *middleware*[26].

Para além do IDE foi também utilizado o **programa de geração de código de inicialização** *STM32CubeMX* da *ST Microeletronics*. Este software gráfico consiste numa ferramenta de configuração que permite a geração de código em C e tem como objetivo reduzir o esforço de desenvolvimento de software de inicialização dos periféricos e consequentemente tempo e custos do projeto. O *STM32CubeMX* tem embebido uma plataforma compreensiva de software (disponível por séries de microprocessadores) que inclui uma biblioteca de abstração de hardware (*STM32Cube HAL*).[27]

No que toca ao software do Router e do Interface do utilizador foi utilizada a linguagem C++ com a *Framework* Qt.

O Qt é uma **framework** *opensource* baseada em C++ com bibliotecas e ferramentas que permitem o desenvolvimento de aplicações multiplataforma, complexas e interativas. A Qt Company (empresa responsável pelo licenciamento e desenvolvimento do Qt) possui também um **IDE** completo (Qt Creator) específico para o desenvolvimento de software em C++, QtScript e QML para todas as plataformas suportadas.[28] Este IDE é bastante simples em aparência mas possui todo um conjunto de ferramentas que auxiliam o rápido desenvolvimento de software tais como o *qmake* que auxilia na criação automático das *makefiles* necessárias para compilar o código gerado, controlo de versões (git e mercurial), possui *debugger* integrado para fazer a depuração de erros tanto local como remotamente, *kits* que permitem fazer a compilação tanto para a máquina em que se está a trabalhar como *cross-compile* para outras arquiteturas, etc. O Qt tem também a vantagem de possuir uma vasta gama de bibliotecas bem documentadas que abrangem praticamente todas as tecnologias mais requisitadas tanto a nível de protocolos de comunicação, como também de interfaces gráficas (QWidgets e QML), *deploy* para o dispositivo na rede e outras bibliotecas de uso mais corrente que facilitam imenso o desenvolvimento (com a garantia de ser um código portável para outras plataformas como Windows, MacOS, Linux,

Andoid e iOS).

Para o software do Servidor o **IDE** utilizado foi o NetBeans por ser uma ferramenta muito completa e modular, podendo ir instalando os *plug-ins* à medida que eles vão sendo necessários. Possui também a vantagem de permitir fazer de forma rápida, via SFTP (*Secure File Transfer Protocol*), o *upload* do código alterado para o servidor e agilizar o processo de desenvolvimento e teste (embora existam outros editores de texto como Sublime e Visual Studio Code capazes de fazer o mesmo). No desenvolvimento deste software, mais importante que a escolha do IDE, é a seleção dos serviços utilizados para que este seja capaz de desempenhar todas as funcionalidades pretendidas. No servidor foi necessário instalar os seguintes **serviços**: Apache, MySQL e No-Ip.

A instalação do *Apache* surge da necessidade de criar um servidor HTTP, sendo este serviço livre e o mais utilizado mundialmente para servir de servidor *web*.

O *MySQL* é um sistema de gestão de base de dados (SGBD) que faz uso da linguagem SQL como interface, sendo um dos mais populares. Este serviço permite a criação e gestão de base de dados para que todos os registos do sistema fiquem guardados num modelo de base de dados relacional que está localizada no servidor.

O *No-Ip* é um serviço que recorre ao DNS de modo a identificar inequivocamente o servidor na rede, sem que para isso seja necessário a atribuição de um IP fixo à máquina (uma vez que esta alternativa se torna muito dispendiosa). Desta forma a máquina pode ser identificada na rede a partir da atribuição de um nome hierárquico. Para além do serviço de DNS é necessário configurar o *router* ao qual o servidor está ligado para permitir a entrada de pedidos do exterior ao servidor, sendo para tal necessário abrir o acesso a determinadas portas e fazer *port forwarding* das mesmas para o servidor.

4.4 Conclusão

Uma vez apresentadas todas as tecnologias de software e hardware que se pretende utilizar na realização do projeto, torna-se necessário detalhar a maneira como estas foram aplicadas. Desta forma, irão ser apresentados no próximo capítulo, mais detalhadamente, os métodos utilizados para a aplicação das tecnologias no que toca ao hardware.

Esta página foi intencionalmente deixada em branco.

Capítulo 5

Hardware

Neste capitulo pretende-se dar ao leitor, uma visão mais aprofundada, da abordagem utilizada para o desenvolvimento do hardware tanto do monitor de infusão, como também do router. Para tal, será necessário dividir o hardware nas suas várias componentes e posteriormente explicar qual o raciocínio utilizado para o desenvolvimento das diversas partes.

5.1 Monitor de Infusão

O monitor de infusão pode ser considerado a parte de maior importância do projeto e sem o qual, este, não faria o mínimo sentido. É nesta parte do projeto que é detetada a frequência ao qual as gotas caem, e desta forma é possível verificar no servidor se a administração se encontra dentro dos valores esperados.

O hardware do monitor de infusão pode ser dividido, para mais facilmente ser analisado nas diferentes partes: alimentação, microcontrolador, XBee e deteção de gotas.

5.1.1 Alimentação

A alimentação do sistema foi pensado para possuir, para além da alimentação normal, a opção de ser alimentada por bateria de modo a não afetar a mobilidade.

No sentido de adicionar uma bateria foi necessário a montagem de um circuito que permitisse o carregamento da bateria quando o circuito está a ser alimentado externamente e por outro lado que a bateria fosse utilizada quando não existe alimentação externa. Existem atualmente circuitos integrados que já são capazes de fazer essa gestão automaticamente, para além de garantir a proteção da bateria. Entre alguns dos produtos desta gama alguns exemplos estudados foram o TPS61090[29], MCP73833[30] e TP4056[31].

Alguns destes integrados já existem em pequenos módulos com todo o circuito necessário, o que facilita imenso na fase de prototipagem. Durante o desenvolvimento deste sistema foi utilizado um módulo que contém o TP4056 (Figura 5.1a). Este módulo é bastante económico e é complementado por um circuito integrado DW01 que protege a bateria de sobrecarregamento, subcarregamento e sobrecorrente (Figura 5.1c).

O pino 5 (Figura 5.1c) do IC TP4056 é o mesmo que carrega a bateria e também alimenta a carga ligada ao OUT+ do módulo (Figura 5.1a). Este pino disponibiliza corrente para a bateria carregar e regulação de tensão de 4,2V[31].

Uma vez que a entrada do circuito já está regulada em tensão a partir do TP4056 e que mesmo quando não tem alimentação externa, a bateria fornece a tensão necessária ao circuito de 3,7V, não sendo necessário adicionar nenhum regulador adicional à saída do módulo. Uma vez que o STM32L100RCT6 tem uma tensão de entrada que deve variar entre os 1,65V e os 3,6V [32], o XBee entre 2.1V e 3.6V e o AMPOP selecionado (LM324) entre os 3V e os 32V, o circuito foi dimensionado para ter uma

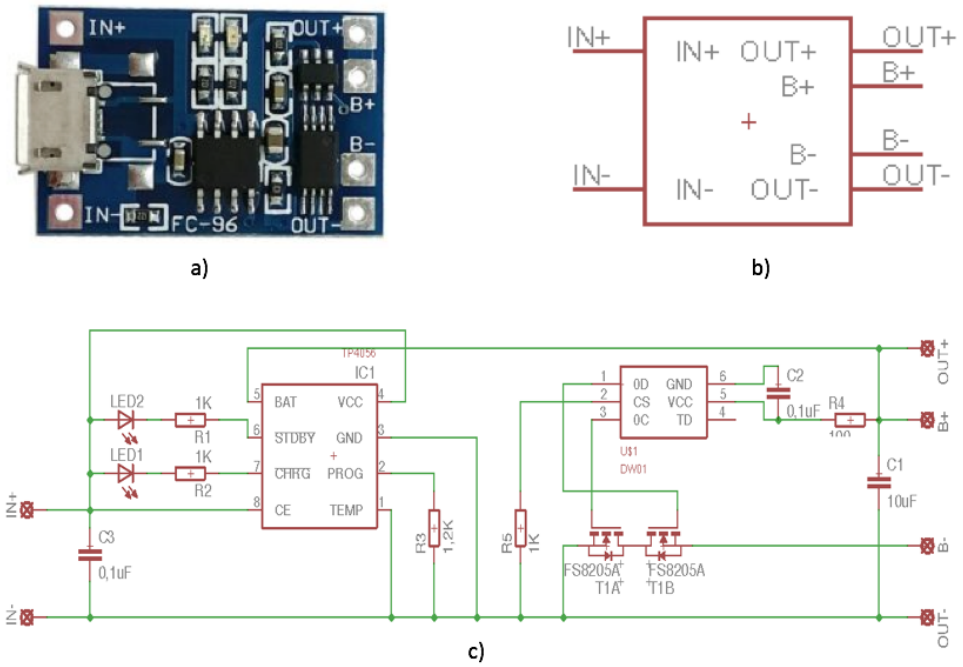


Figura 5.1: a) Módulo TP4056; b) Símbolo no esquemático do módulo; c) Circuito do módulo TP4056.

tensão de entrada entre os 3V e os 3,6V, uma vez que é uma gama de tensão que todos os dispositivos presentes no circuito suportam sem subalimentar nem sobrealimentar os mesmos.

Para fazer a adaptação da tensão não se utilizou outro regulador para além do interno do TP4056, pois geralmente requerem uma diferença de tensão entre a entrada e a saída superior à disponível neste caso ($3,7V - 3,6V = 0,1V$). Em vez de um regulador, optou-se por utilizar um díodo de Zener com uma tensão de rutura de 3,3V uma vez que é um dos valores *standard* e que garante que a tensão fica dentro da gama e não excede nenhum limite.

Como resultado final de todo o estudo foi desenvolvido o circuito apresentado na Figura 5.2.

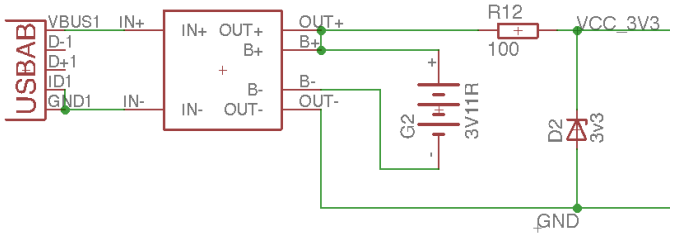


Figura 5.2: Circuito de alimentação do Monitor de Infusão

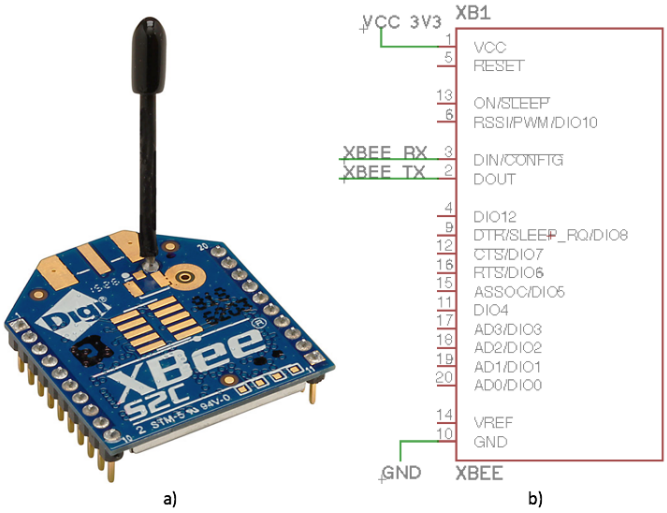


Figura 5.3: a) Módulo XBee utilizado; b) Circuito de ligação com o XBee

5.1.2 XBee

Tal como falado na secção 4.2.2 o módulo utilizado para implementar o protocolo ZigBee foi o XBee. O XBee para além de facilitar a implementação do protocolo de comunicação torna-se bastante simples no que toca às ligações físicas ao módulo.

O módulo XBee escolhido (XB24CZ7WIT-004)(Figura 5.3a) é de montagem discreta e possui 20 pinos com funções diferentes (Figura 5.4), sendo algumas configuráveis como entradas digitais, analógicas, utilização de SPI para comunicação com micro-controlador ou mesmo como pinos de controlo da comunicação UART.

Para a montagem em questão foram apenas necessários utilizar os pinos

Pin	Name	Direction	Function
1	VCC	-	Power supply
2	DOUT	Output	UART data out
3	DIN/$\overline{\text{CONFIG}}$	Input	UART data In
4	SPI_MISO	Output	Serial Peripheral Interface (SPI) Data Out
5	$\overline{\text{RESET}}$	Input	Module reset (reset pulse must be at least 200 ns). This must be driven as an open drain/collector. The device drives this line low when a reset occurs. Never drive this line high.
6	PWM0/RSSI PWM	Output	PWM output 0 / RX signal strength indicator
7	PWM1	Output	PWM output 1
8	[Reserved]	-	Do not connect
9	DI8/$\overline{\text{SLEEP}}$_ RQ/DTR	Input	Pin sleep control line or digital input 8
10	GND	-	Ground
11	DIO4/SPI_MOSI	Both	Digital I/O 4 / SPI Data In
12	DIO7/$\overline{\text{CTS}}$	Both	Digital I/O 7 / Clear-to-send flow control
13	ON/$\overline{\text{SLEEP}}$	Output	Device sleep status indicator
14	V_{REF}	-	Feature not supported on this device. Used on other XBee devices for analog voltage reference.
15	DIO5/ASSOC	Both	Digital I/O 5 / Associated indicator
16	DIO6/$\overline{\text{RTS}}$	Both	Digital I/O 6 / Request-to-send flow control
17	$\overline{\text{DIO3}}$/AD3/SPI_ SSEL	Both	Digital I/O 3 / Analog input 3 / SPI select
18	DIO2/AD2/SPI_ CLK	Both	Digital I/O 2 / Analog input 2 / SPI clock
19	$\overline{\text{DIO1}}$/AD1/SPI_ ATTN	Both	Digital I/O 1 / Analog input 1 / SPI Attention
20	DIO0/AD0	Both	Digital I/O 0 / Analog input 0

Figura 5.4: Descrição dos pinos XBee.[7]

de alimentação, DIN e DOUT (Figura 5.3b) uma vez que a interface de comunicação utilizada foi UART e os pinos DIN e DOUT correspondem, respetivamente, aos pinos RX e TX de uma comunicação série deste género.

A utilização deste módulo reside também na sua tensão de funcionamento que é perfeitamente compatível com a do microcontrolador, não sendo, portanto, necessário haver nenhum tipo de circuito de *level-shifting*.

5.1.3 Detetor de gotas

Nesta porção de hardware pretende-se fazer a deteção da passagem das gotas através da interrupção de um feixe luminoso. Para tal foi necessário encontrar um par emissor-recetor que operasse dentro de uma gama de radiação semelhante para se poder detetar alterações à radiação imposta pelo emissor.

Para o emissor foi escolhido um LED com boa luminescência (para que o restante circuito não seja tão sensível a interferência externa) e que suporta uma corrente até 60mA. De modo a dimensionar o circuito do emissor, e não aumentar muito o consumo energético, foi feito o cálculo do valor da resistência para uma corrente de 40mA, tensão de alimentação de 3,3V e queda de tensão no LED de 0,7V através da equação 5.1, o que resultou no esquema da Figura 5.5. O valor da resistência aplicado no circuito, corresponde ao valor de mercado mais próximo do valor resultante da equação que neste caso será 68Ω, o que resulta num desvio que quase pode ser desprezável em relação ao definido inicialmente.

$$\begin{aligned} V = I * R \iff R = \frac{V}{I} \iff R = \frac{V_{CC} - V_{LED}}{I} \\ \iff R = \frac{3,3V - 0,7V}{40mA} \iff R = 65\Omega \approx 68\Omega \end{aligned} \tag{5.1}$$

Para o recetor foi escolhido o fototransístor OFT-5301. Este tipo de componentes funciona de modo semelhante a um transístor, no entanto a

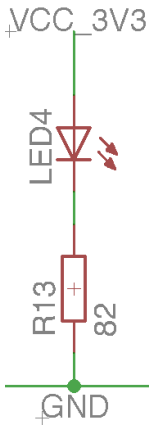


Figura 5.5: Esquemático do emissor do sensor de gotas

base é sensível à radiação. Para a montagem do fototransístor utilizou-se uma resistência que foi dimensionada de acordo com a alimentação, tensão entre coletor-emissor e a corrente no coletor no escuro. Para o fototransístor em questão a tensão coletor-emissor é 0,3V e a corrente no escuro é de 100nA, sendo portanto o valor da resistência (dada pela equação 5.2) de 30kΩ. No entanto este valor não é um dos valores de resistência normalizados por isso foi utilizado o valor seguinte mais próximo de 47kΩ, assim como é possível observar na Figura 5.6a.

$$R = \frac{VCC - V_{CE}}{If} \rightarrow R = \frac{3.3V - 0,3V}{0,0001A} \longleftrightarrow R = 30k\Omega \tag{5.2}$$

Numa primeira fase do condicionamento de sinal, foi adicionado ao circuito um filtro passa-alto de maneira a filtrar a componente continua do sinal (resultante da radiação incidente e pequenas alterações na luz ambiente) e apenas detetar as perturbações do sinal. Para tal, foi necessário ter em consideração qual é a resposta do sensor à passagem de uma gota pelo feixe de luz, de maneira a saber quanto tempo o sinal do sensor é perturbado e garantir que passa no filtro passa-alto.

Após uma pequena simulação do processo normal foi detetado que a gota

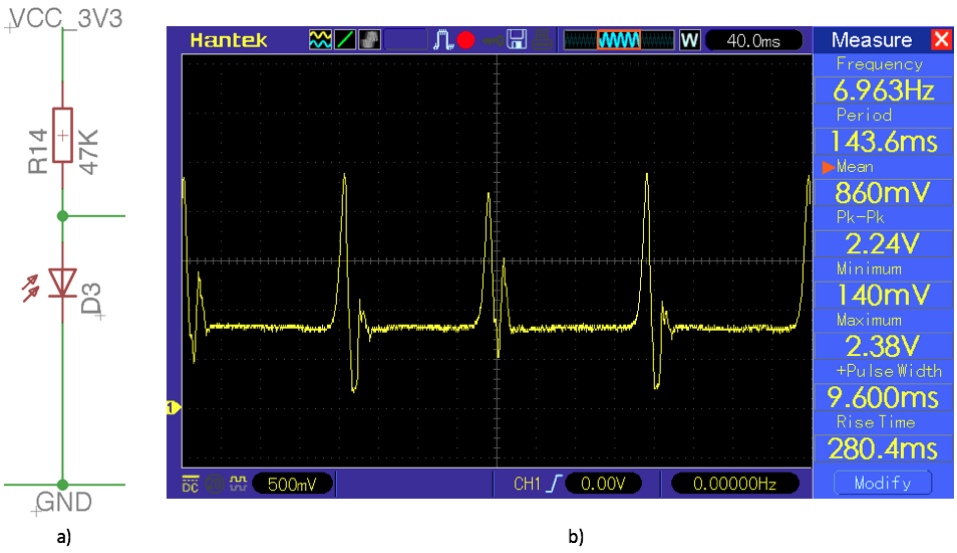


Figura 5.6: a) Esquemático do recetor do sensor de gotas; b) Sinal resultante da montagem base do recetor.

tem a influência no sinal representado na Figura 5.6b. Nesta amostragem é possível verificar a alteração significativa do sinal durante aproximadamente 50ms e que este apresenta dois picos (resultantes da refração da luz durante a queda da gota), o que resulta num sinal com um período de 25ms (40Hz).

De modo a dimensionar o filtro para filtrar pequenas alterações à luz incidente de maneira constante sem, no entanto, perturbar o sinal que se pretende tratar, definiu-se como frequência de corte (f_c) os 10Hz. Para efetuar o dimensionamento do circuito utilizou-se a equação 5.3, onde se definiu a frequência de corte e um valor do condensador. A partir do valor da resistência obtido efetuou-se um arredondamento para o valor comercial mais próximo, o que resultou numa frequência de corte de 10,61Hz (equação 5.4).

$$f_c = \frac{1}{2 * \pi * R * C} \rightarrow 10Hz = \frac{1}{2 * \pi * R * 15nF} \longleftrightarrow$$
$$\longleftrightarrow R = \frac{1}{10 * 2 * \pi * 15nF} \longleftrightarrow R \approx 1,061M\Omega \approx 1M\Omega \tag{5.3}$$

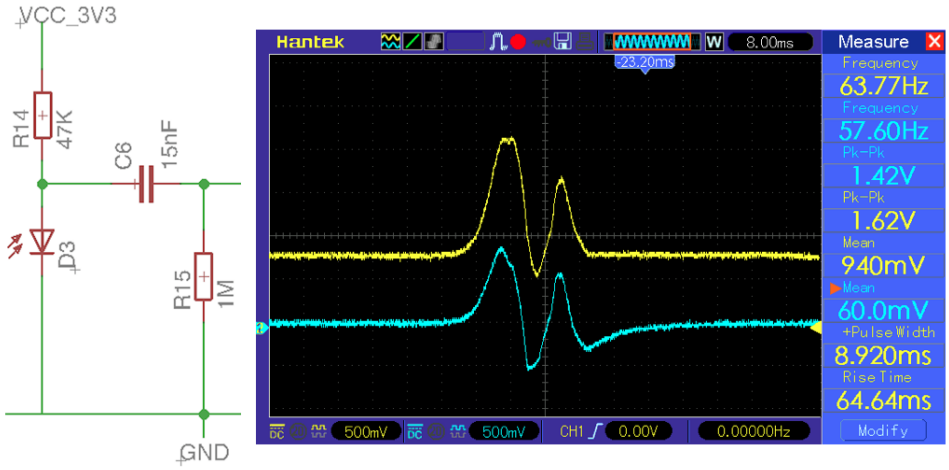


Figura 5.7: a) Esquemático do recetor do sensor de gotas com filtro passa-alto; b) Sinal resultante da montagem do recetor com filtro passa-alto (a azul) vs montagem do recetor sem filtro passa-alto (a amarelo).

$$f_c = \frac{1}{2 * \pi * R * C} \rightarrow f_c = \frac{1}{2 * \pi * 1M\Omega * 15nF} \longleftrightarrow f_c \approx 10,61Hz \quad (5.4)$$

Como resultado final deste dimensionamento resulta o circuito presente na Figura 5.7a. Deste circuito obtém-se a filtragem da componente contínua como é possível verificar na Figura 5.7, onde a média da tensão passa de 940mA para 60mA.

Uma vez que os pinos do microcontrolador, teoricamente, detetam nível lógico alto aos 1,887V (equação 5.5)[32] e com o circuito atualmente desenvolvido para o propósito pode ser verificado (na Figura 5.7) que a tensão à saída deste é inferior a 1V é, portanto, necessário proceder à amplificação do mesmo de maneira a garantir que a tensão de saída do circuito esteja entre os 1,887V e os 3,3V.

$$V_{IH} = 0,39 * VDD + 0,59 \rightarrow V_{IH} = 0,39 * 3,3 + 0,59 = 1,887V \quad (5.5)$$

Para proceder à amplificação do sinal foi utilizado o AMPOP LM324[33] em montagem não inversora. Assumindo o pior cenário, onde o sinal da gota apenas atinge 0,5V e se pretende atingir uma tensão entre 1,887V e 3,3V, o ganho necessário está entre 3,8(equação 5.6) e 6,6(equação 5.7). Uma vez que a alimentação do AMPOP é a mesma que a do microcontrolador a sua saída irá no máximo ser igual ao valor da sua alimentação (no máximo 3,3V). Desta forma, não haverá o problema de poder danificar o pino de entrada do microcontrolador e mesmo que o ganho do AMPOP seja superior ao necessário, o sinal à saída irá saturar e não irá ter um valor superior à tensão de alimentação.

$$G_{min} = \frac{1,887}{0,5} = 3,774 \approx 3,8 \tag{5.6}$$

$$G_{max} = \frac{3,3}{0,5} = 6,6 \tag{5.7}$$

De modo a garantir que o ganho não irá amplificar o ruído existente para provocar erros de leitura, definiu-se um ganho que está localizado entre os limites acima definidos de 6, usando o circuito utilizado na Figura 5.8a que foi calculado a partir da equação 5.8. Desta forma, o valor aplicado está dentro dos limites previstos , tendo sido definido o valor de uma resistência (1kΩ) e o outro calculado e aproximado para um valor onde comercialmente seja mais fácil encontrar. No final, verificou-se que os valores aproximados ainda se encontravam dentro dos limites (equação 5.9).

$$\begin{aligned} V_o = (1 + \frac{R17}{R16})V_i \longleftrightarrow \frac{V_o}{V_i} = (1 + \frac{R17}{R16}) \rightarrow 6 = (1 + \frac{R17}{1k}) \longleftrightarrow \\ \longleftrightarrow 6k = 1k + R17 \longleftrightarrow R17 = 5k \approx R17 = 4,7k \end{aligned} \tag{5.8}$$

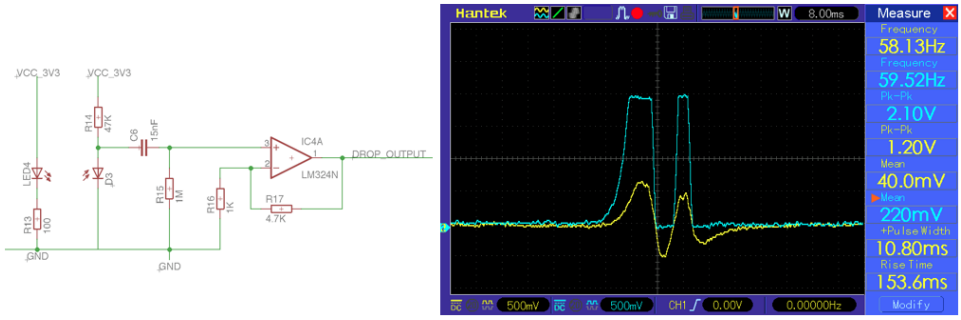


Figura 5.8: a) Esquemático do sensor completo de gotas; b) Sinal resultante do sensor completo de gotas (a azul) vs montagem do recetor com filtro passa-alto (a amarelo).

$$V_o = (1 + \frac{R17}{R16})V_i \longleftrightarrow \frac{V_o}{V_i} = (1 + \frac{R17}{R16}) \rightarrow$$
$$\rightarrow \frac{V_o}{V_i} = (1 + \frac{4,7k}{1k}) \longleftrightarrow \frac{V_o}{V_i} = 5,7 \tag{5.9}$$

Como resultado final da aplicação do circuito da Figura 5.8a, pode-se observar uma resposta do sinal na Figura 5.8b onde a amplificação do sinal é limitada superiormente e inferiormente pela tensão de alimentação do AMPOP (que é a mesma que utilizada para alimentar o micro-controlador).

5.1.4 Microcontrolador

O microcontrolador é o núcleo da parte do monitor de infusão. É nesta parte do hardware que está centrado todo o processamento da informação proveniente do detetor de gotas e XBee, assim como enviar informação para outros elementos do sistema através do XBee.

Embora o microcontrolador seja o elemento que vai servir para unir todos os periféricos, não funciona sem a existência de alguns circuitos auxiliares que permitem fazer o *reset* ao dispositivo Figura 5.9a, assim como descarregar o programa para a memória do microcontrolador Figura 5.9b.

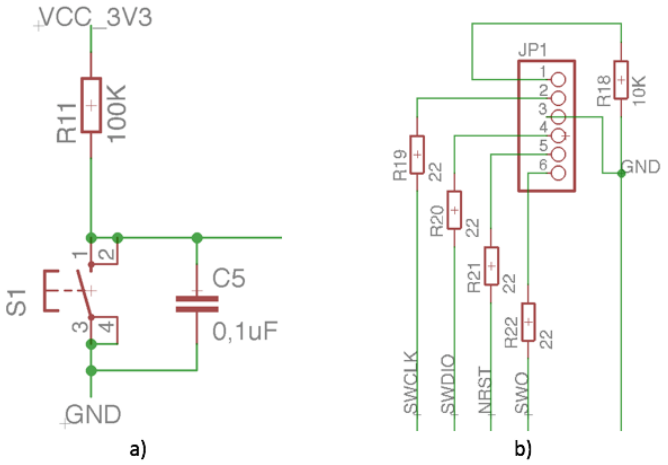


Figura 5.9: a) Circuito de *reset* do microcontrolador; b) Circuito de programação e *debug*.

No caso do circuito de *reset* do microcontrolador foi necessário ter em atenção que o pino (7) de *reset* do STM32L100RCT requer nível lógico alto para o normal funcionamento e nível lógico baixo para que seja feito o *reset* do mesmo. Para tal foi colocada uma resistência de *pull-up* e foi adicionado um botão para que fosse possível efetuar o *reset* do mesmo manualmente.

Para o circuito que permite efetuar a programação e *debug* do microcontrolador apenas foi necessário seguir as recomendações do fabricante, de modo a ser possível fazer a programação a partir do dispositivo ST-LINK/V2 (programador e *debugger* oficial da STMicroeletronics).

De seguida foi também necessário estabelecer a ligação entre o detetor de gotas e o microcontrolador. Para tal foi escolhido o pino PA8 que é tolerante a uma entrada de 5V e pode também ser configurado para funcionar por interrupção externa.

5.1.5 Resultado Final

Como resultado final de todas as partes que integram o monitor de infusão (e foram detalhadas em cima) pode-se observar o circuito que se encontra representado na Figura 5.10.

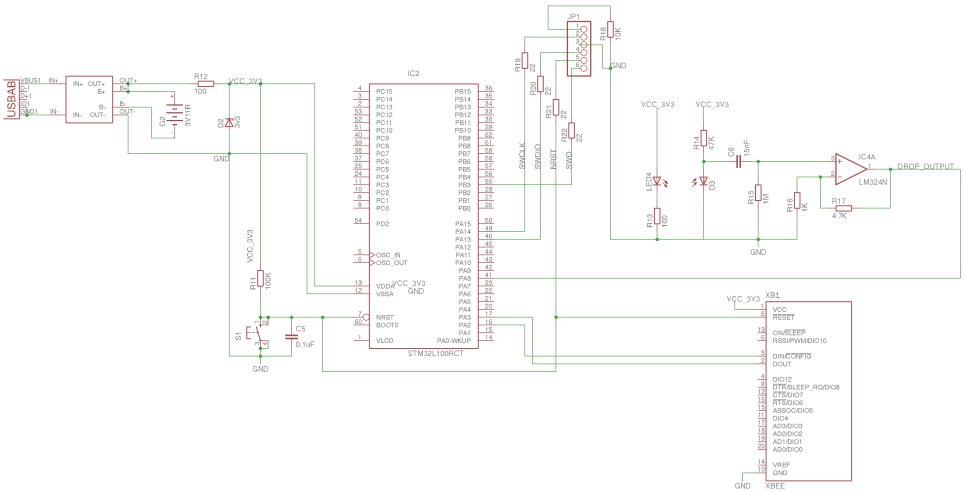


Figura 5.10: Circuito do Monitor de Infusão

5.2 Router

O router neste sistema desempenha o papel de encaminhador de mensagens entre o monitor de infusão e o servidor. Para esse efeito, pretendeu-se fazer um hardware que fosse simples e do qual pudesse ser aplicado a um dispositivo com ligação à internet e uma porta USB disponível.

Tal como o monitor de infusão envia informação a partir de um XBee, o router também necessita de um XBee para receber a informação que pretende tratar para, posteriormente, enviar para o servidor. O modo que o XBee utiliza para enviar uma trama recebida é por UART (tal como explicado na secção 5.1.2).

Para fazer a transmissão das tramas recebidas, para estas serem tratadas num dispositivo que se pretende que tenha ligação à internet e um USB *host*, foi necessário a utilização de hardware capaz de fazer a tradução do sinal TTL da UART para um USB que seja reconhecido numa máquina como uma porta série. De modo a fazer esta tradução de protocolos foi utilizada uma FTDI, que mais especificamente está inserida num módulo (Figura 5.11) com o *pinout* do XBee. Este módulo facilitou em muito o desenvolvimento

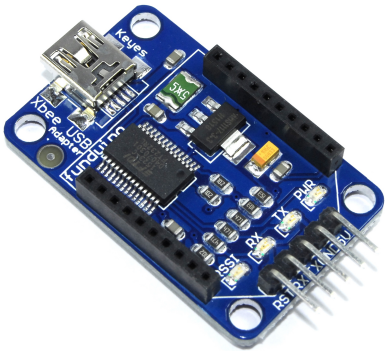


Figura 5.11: Módulo com FTDI para emulação de porta série USB a partir de TTL UART

do router e passou o resto da funcionalidade e tratamento de informação para o software, sendo apenas necessário acoplar o XBee ao módulo.

Para a unidade de processamento da informação foi utilizado um computador normal (onde se ligou o USB do módulo), no entanto também se poderia utilizar uma pequena placa de desenvolvimento (como por exemplo um Raspberry Pi).

Uma vez detalhado todo o hardware utilizado neste projeto é, portanto, necessário passar para o desenvolvimento do software, que está por trás do hardware, que faz todo o processamento da informação recolhida, assim como a gestão/criação/consulta de todos os registos gerados pelo sistema. No próximo capítulo irá tratar-se, portanto, toda esta problemática.

Capítulo 6

Software

Neste capitulo pretende-se começar a dar uma noção do fluxo de informação ao longo do sistema a partir do Monitor de Infusão e da Interface com o utilizador. De seguida será aumentado o detalhe da modulação do código e a fazer a descrição da solução utilizada de cada um dos elementos constituintes do sistema.

No que toca ao fluxo da informação ao longo do sistema, este pode ser desencadeado por dois elementos diferentes, nomeadamente pelo Monitor de Infusão e pela Interface com o Utilizador. No caso do Monitor de Infusão, este apenas envia mensagens para reportar as leituras feitas ao longo de um minuto e segue o fluxo apresentado na Figura 6.1. O Interface com o Utilizador envia vários tipos de mensagens, nomeadamente de inserção, alteração e consulta de registos no entanto o fluxo de informação segue todo a mesma lógica, podendo portanto ser generalizado pelo diagrama de sequência apresentado na Figura 6.2.

6.1 Monitor de Infusão

O monitor de infusão é a parte do sistema que faz a monitorização da taxa de administração e periodicamente envia o estado atual para o Router.

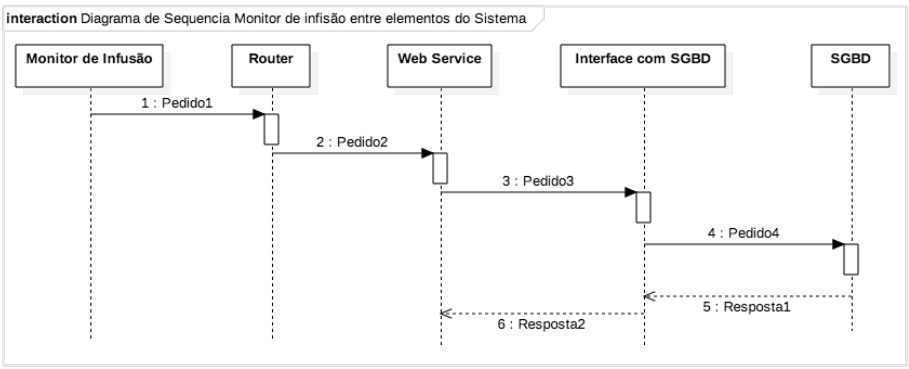


Figura 6.1: Diagrama de sequência do fluxo de informação do Monitor de Infusão ao longo do sistema.

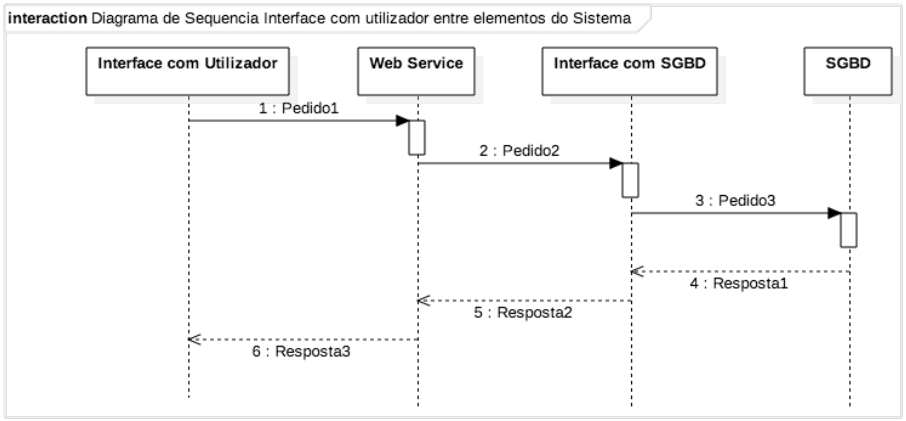


Figura 6.2: Diagrama de sequência do fluxo de informação do Interface com o Utilizador ao longo do sistema.

Description	Command	Value					
		0	1	2	3	4	5
Report Value	1	Medication read		Medication set		Medication margin	alarm

Figura 6.3: Estrutura da trama *Report Frame* para reportar a taxa detetada.

6.1.1 Análise de Requisitos

O monitor de infusão deve, no seu funcionamento, fazer o processamento do sinal proveniente do hardware desenvolvido para fazer a deteção das gotas (secção 5.1.3), filtrar este sinal para evitar a deteção de ruído ou "falsas gotas"e periodicamente (minuto a minuto) enviar a informação da taxa detetada para o Router de maneira a que este possa fazer a inserção do registo na Base de Dados.

6.1.2 Estrutura das tramas

Para efetuar a transmissão dos valores de medicação detetados para o Router (mensagem 1 da Figura 6.1), foi utilizada uma trama com uma estrutura representada na Figura 6.3. Nesta trama é possível observar que existem cinco campos diferentes, nomeadamente o campo *Command*, *Medication Read*, *Medication Set*, *Medication Margin* e *Alarm*. O campo *Command* é obrigatório em todas as tramas pois permite identificar que tipo de trama e a partir desta conhece a sua estrutura e a maneira de a processar, o campo *Medication Read* contém a taxa de medicação detetada pelo microcontrolador, o campo *Medication Set* possui o valor para o qual foi programada o valor da medicação, o campo *Medication Margin* possui a margem/ desvio (em percentagem) que o valor programado pode ter antes de ser considerado alarme e por fim o campo *Alarm* indica se o valor lido deve ser interpretado como anomalia ou não. Apesar destes campos estarem todos estabelecidos na trama, apenas o *Command* e *Medication Read* são realmente utilizados e a responsabilidade de calcular o desvio e gerar o alarme fica no servidor, servindo portanto os restantes campos para trabalho futuro.

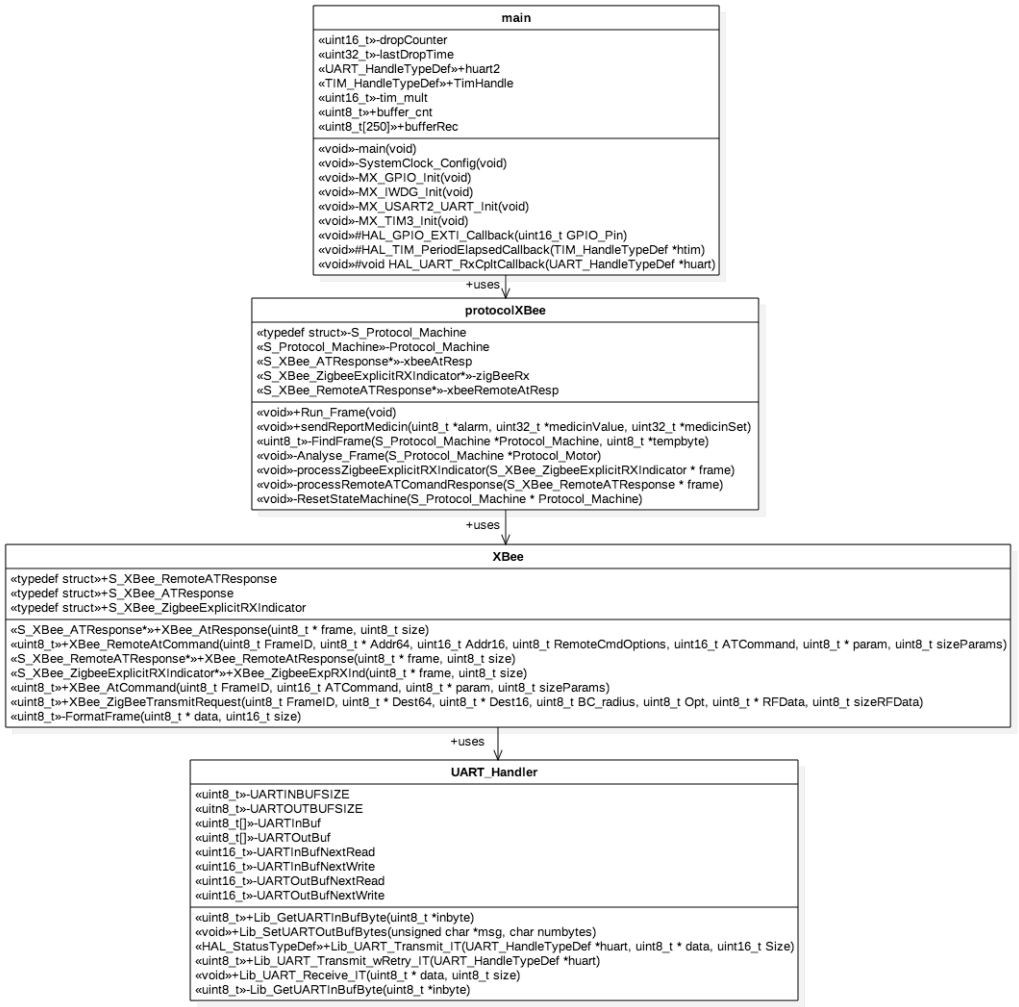


Figura 6.4: Diagrama de classes do Monitor de Infusão

6.1.3 Modulação de Código

A programação desenvolvida para o monitor de infusão foi estruturada de modo a tentar compartimentar o programa, sendo assim possível ir criando uma camada de abstração à medida que se vai subindo o nível da programação. Nesse sentido foi criada a estrutura apresentada no diagrama de classes da Figura 6.4.

Começando do baixo para o alto nível temos em primeiro lugar a classe

UART_Handler. Esta tem como principal objetivo criar uma interface entre o utilizador e o periférico UART. A *UART_Handler* faz uso de dois *buffers* circulares. Estes podem ser preenchidos a partir da utilização da função *Lib_SetUARTOutBufBytes* para adicionar um determinado numero de bytes ao *buffer* de saída e a função *Lib_UART_Receive_IT* para preencher o *buffer* de entrada. Para além de preenchidos o *buffer* de entrada pode ser consumido utilizando a função *Lib_GetUARTInBufByte* que retorna um byte do *buffer*. O *buffer* de saída deve apenas ser preenchido com a informação pretendida e de seguida deve ser chamada a função *Lib_UART_Transmit_IT* que irá enviar a informação pela UART configurada por interrupção, ou então usando *Lib_UART_Transmit_wRetry_IT* que faz um procedimento semelhante à função anterior, no entanto cada byte pode sofrer várias tentativas de envio até ser bem-sucedido.

Ao subir um nível na hierarquia do software temos a classe *XBee*. Esta camada é responsável por criar a camada de abstração da estruturação das diferentes tramas XBee, sendo apenas necessário fazer a chamada às funções que correspondem às tramas que se pretende enviar e passar os devidos parâmetros. Da mesma maneira que pode ser feito o encapsulamento da trama, é também possível fazer o desencapsulamento. Para isso, é necessário detetar que tipo de trama foi encontrada e enviar o resto em bruto como argumento, sendo que é retornado uma estrutura com todos os campos da trama descodificados numa estrutura para mais fácil manuseamento. Esta camada faz uso da *UART_Handler* para usada a porta UART.

Ao nível do *protocolXBee* está embebida toda a camada de negócio da informação que chega e sai do barramento do módulo XBee. A principal função que faz desencadear todos os outros eventos é a *Run_Frame* uma vez que é nesta que são consumidos todos os dados que chegam pela porta UART através de uma máquina de estados, onde são procuradas tramas válidas, e quando encontradas são tratadas pela função *Analyse_Frame*. É também

nesta camada que é criada a camada de abstração, para o nível mais alto, da constituição das tramas que são enviadas para o Router, como é o caso da função *sendReportMedicin* que faz a formatação dos valores passados por parâmetro e de seguida utiliza a função *XBee_ZigBeeTransmitRequest* da camada inferior para fazer o encapsulamento numa trama XBee.

No que toca ao nível superior, é neste que são recebidas as interrupções dos periféricos do microcontrolador tais como interrupção de GPIO, Temporizador e receção de dados na UART. Para além destas funcionalidades, esta camada também é responsável por manter o *watchdog* ativo e a máquina de processamento de tramas a funcionar.

6.1.4 Descrição de Solução

O funcionamento deste software rege-se por duas categorias diferentes, onde cada uma tem o seu papel fundamental nomeadamente as interrupções dos periféricos e o processamento do buffer de entrada da UART.

As ações tomadas nas interrupções do sensor de gotas, do timer utilizado para reportar periodicamente os valores e a interrupção da UART por causa da receção de dados no buffer podem ser observadas nas Figuras 6.5, 6.6 e 6.7, respetivamente.

A interrupção do sensor de gotas indica um pico de tensão à saída do sensor, no entanto nem todos estes picos podem corresponder a gotas uma vez que (tal como visto na Figura 5.8b da secção 5.1.3), uma única gota pode provocar dois picos de tensão devido à refração da luz. A juntar a este efeito é possível filtrar as gotas verdadeiras do ruído, uma vez que sabemos que as gotas têm uma duração média de 50ms e que (por experiência prática) nunca foi detetada uma frequência de queda superior a 10Hz (intervalo de ≈100ms entre gotas) é possível filtrar os impulsos válidos dos inválidos. Após a passagem por este filtro apenas é necessário incrementar o contador de gotas. Todo o algoritmo (Figura 6.5) está codificado segundo o Código

6.1.

Código 6.1: Interupção do *GPIO*.

```
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
  unsigned char button_state=0;
  // Drop Sensor
  if( GPIO_Pin == GPIO_PIN_8)
  {
      if(HAL_GPIO_ReadPin(GPIOA,GPIO_PIN_8)==GPIO_PIN_SET)
      {
          if(HAL_GetTick() - lastDropTime > DROP_WIDTH_MS)
          {
              dropCounter++;
              lastDropTime = HAL_GetTick();
              HAL_GPIO_TogglePin(GPIOC,GPIO_PIN_8);
          }
      }
  }
}
```

A interrupção do final de temporização indica o momento em que um novo valor deve ser enviado para registo na base de dados. No final de cada temporização, é calculada a taxa de gotas que passaram pelo sensor num minuto e é enviada a trama. No caso da temporização ser diferente de um minuto é necessário fazer a proporção para encontrar o valor correto a enviar, caso contrário não é necessário conversão. Neste projeto foi definida uma temporização de um minuto porque a medicação nunca poderá logicamente ser inferior a 1gota/min, desta forma a informação tem uma cadência de atualização razoável para a aplicação em questão, não se insere uma grande carga de tráfego na rede ZigBee e consegue-se aumentar também a autonomia da bateria (ao não realizar demasiadas transmissões

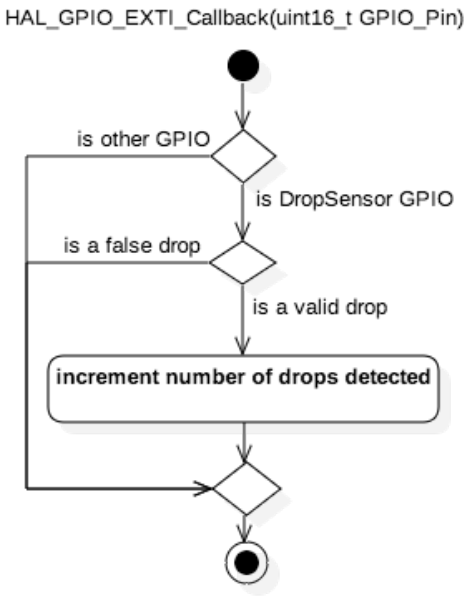


Figura 6.5: Interrupção externa do GPIO do sensor de gotas.

pelo XBee). Uma vez que o XBee no monitor de infusão está configurado como Endpoint (RFD) de uma rede ZigBee, este apenas pode conectar-se a um dispositivo do tipo FFD (Router ou Coordenador de rede) o que facilita imenso a tarefa de enviar uma trama para reportar os valores, uma vez que não é necessário elaborar um protocolo de mensagens trocadas com a rede para descobrir qual o endereço do Router ao qual está ligado mas basta enviar uma trama do tipo *Transmit Request* como *broadcast* mas com o raio de propagação a um, garantindo-se desta forma que a trama apenas é recebida pelo Router ao qual está ligado não aumentando desta forma o tráfego na rede. O código utilizado está representado no Código 6.2 que tem o seu algoritmo representado no diagrama de sequencia da Figura 6.6.

Código 6.2: Interupção do *timer*.

```
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
{
  //check alarm
  char medication_alarm = 0x00;
  uint16_t medMarginInDrops=(medication_set*medication_margin)/100;
  if ( dropCounter < medication_set - medMarginInDrops ||
      dropCounter > medication_set + medMarginInDrops)
  {
    medication_alarm = 0x01;
  }
  uint8_t bufferParams[64];
  //Report command
  bufferParams[0] = 0x01 ;
  //medication value read
  bufferParams[1] = dropCounter & 0xFF ;
  bufferParams[2] = (dropCounter >> 8) & 0xFF ;
  //medication value set
  bufferParams[3] = medication_set & 0xFF;
  bufferParams[4] = (medication_set >> 8) & 0xFF ;
  //medication margin set
  bufferParams[5] = medication_margin & 0xFF;
  //medication value set
  bufferParams[6] = medication_alarm & 0xFF;
  uint8_t bufferMAC[8];
  //send broadcast message
  for(int i=0;i<8;i++)
  {
    bufferMAC[i]=0;
  }
  bufferMAC[6]=0xFF;
  bufferMAC[7]=0xFE;
  XBee_ZigBeeTransmitRequest(0x01,bufferMAC,0x0000,0x01,0x00,bufferParams,7);
```

```
  dropCounter = 0;
}
```

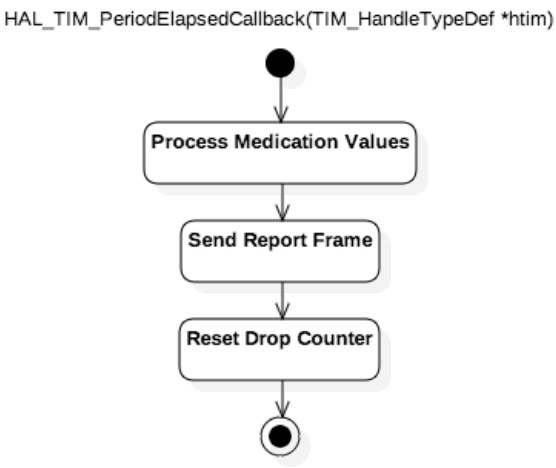


Figura 6.6: Interrupção de *timout* do Timer usado para reportar os valores lidos.

No que toca à interrupção de receção da UART a ação tomada é simplesmente guardar a informação recebida num buffer circular presente na classe UART_Handler para que posteriormente possa ser consumido durante o ciclo principal, tal como representado no diagrama da Figura 6.7 e no Código 6.3.

Código 6.3: Interupção da UART.

```
void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart)
{
  if(huart->Instance==USART2)
  {
    buffer_cnt+=huart->RxXferSize;
    Lib_UART_Receive_IT(bufferRec,buffer_cnt);
    buffer_cnt=0;
  }
```

```
}
```

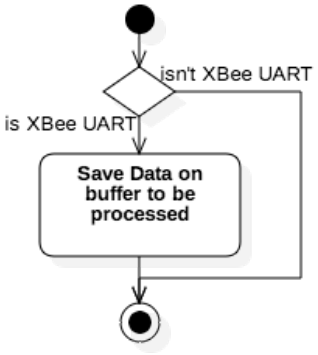


Figura 6.7: Interrupção da UART por receção de dados.

A função *main* tem um papel fundamental pois em primeiro lugar é responsável pela inicialização de todos os periféricos e depois é durante o ciclo infinito que verifica se existem novos dados no buffer da UART e é processada a informação recebida que foi guardada em buffer (Figura 6.8).

O *Process buffer* (Figura 6.9) da *main*, é um processo composto por várias partes gerais, nomeadamente a preparação da máquina de estados que procura a trama, a procura e validação de tramas contidas no buffer e o processamento da trama em si. Na primeira fase verifica-se se a máquina de estados responsável por encontrar tramas válida no buffer poderá em algum momento estar bloqueada em algum ponto e caso esteja executa-se um "*watchdog*"que irá reiniciar a máquina de estados. Ainda dentro da primeira parte é retirado um byte do buffer circular de entrada para que este seja processado na máquina de estados. Na segunda parte, a máquina de estados corre com o byte retirado e é verificado se este pertence a uma trama. Assim que a máquina de estados encontra o último byte de uma trama, é assinalada a presença de uma trama válida e esta é guardada numa estrutura. Por fim, a trama encontrada é analisada e conforme a mensagem/comando que contenha, esta pode ter diferentes ações. Todo o fluxo

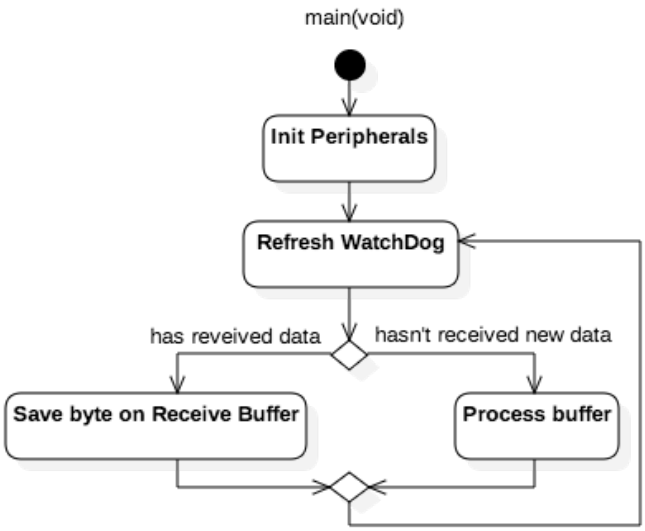


Figura 6.8: Diagrama de atividade da função *main* do monitor de infusão.

descrito está representado no Código 6.4.

Código 6.4: Processamento do buffer.

```
void Run_Frame(void)
{
  uint8_t tempbyte;
  //Timout to prevent a frame became lost on state machine.
  if(HAL_GetTick() > MyGlobalTime + 500)
  {
    ResetStateMachine(&Protocol_Machine);
    MyGlobalTime = HAL_GetTick();
  }
  //Get and process the recived frame
  if (Lib_GetUARTInBufByte(&tempbyte) != 0)
  {
    if(FindFrame(&Protocol_Machine, &tempbyte))//verify if found a
        valid frame
```

```
    {
      Analyse_Frame(&Protocol_Machine); //Process the valid frame
    }
  }
}
```

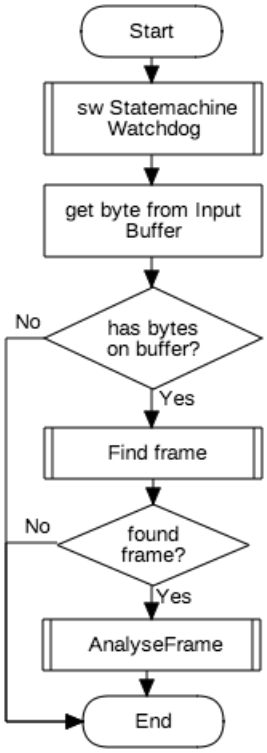


Figura 6.9: Fluxograma que descreve o Process buffer da main

6.2 Router

O Router é a parte do sistema responsável por fazer de nó da rede ZigBee ao qual se conectam os monitores de infusão e depois de fazer a receção de tramas do monitor de infusão, este terá de fazer o pedido ao servidor conforme a trama que receber.

6.2.1 Análise de Requisitos

Durante o seu funcionamento, o Router, deve ser capaz de entender todas as tramas provenientes do monitor de infusão. Para além destas, também tem de ser capaz de saber fazer os pedidos ao servidor consoante os comandos recebidos, isto é, para além de ter de conhecer todas as tramas que o monitor de infusão envia, tem de conhecer a constituição dos pedidos para o servidor.

6.2.2 Modulação do Código

Durante o desenvolvimento da programação para o software do Router, foram utilizados no C++ diferentes classes com o objetivo de separar as funcionalidades em módulos mais simples. No final a estruturação do código resultou no diagrama de classes representado na Figura 6.10.

A *XSerialPort* foi a classe de mais baixo nível deste software que permite aceder e interagir com a porta série. Para tal, esta classe utiliza outras classes nativas da *framework* Qt, nomeadamente a classe QSerialPort para aceder à porta série e a QSerialPortInfo para obter informações sobre as portas séries disponíveis. A partir do uso das funcionalidades nativas do Qt é, portanto, possível configurar e utilizar a porta série.

A classe *xbeeDriveUtils* foi desenvolvida para encontrar as tramas no buffer de receção. Para fazer a busca da trama, esta classe, utiliza uma máquina de estados muito semelhante à utilizada no monitor de infusão, apenas sendo adaptada para tirar proveito das características do C++, uma vez que a trama recebida é do XBee e, portanto, igual nestes dois dispositivos.

Na classe *xbeeFrameFactory* é possível fazer o desencapsulamento da informação da trama encontrada pela *xbeeDriveUtils*. Deste processo resulta um sinal (especifico de cada um dos tipos das diferentes tramas XBee) que pode ser tratado num nível superior. Nesta parte também é possível fazer a formatação de uma trama XBee, sendo apenas necessário indicar qual o

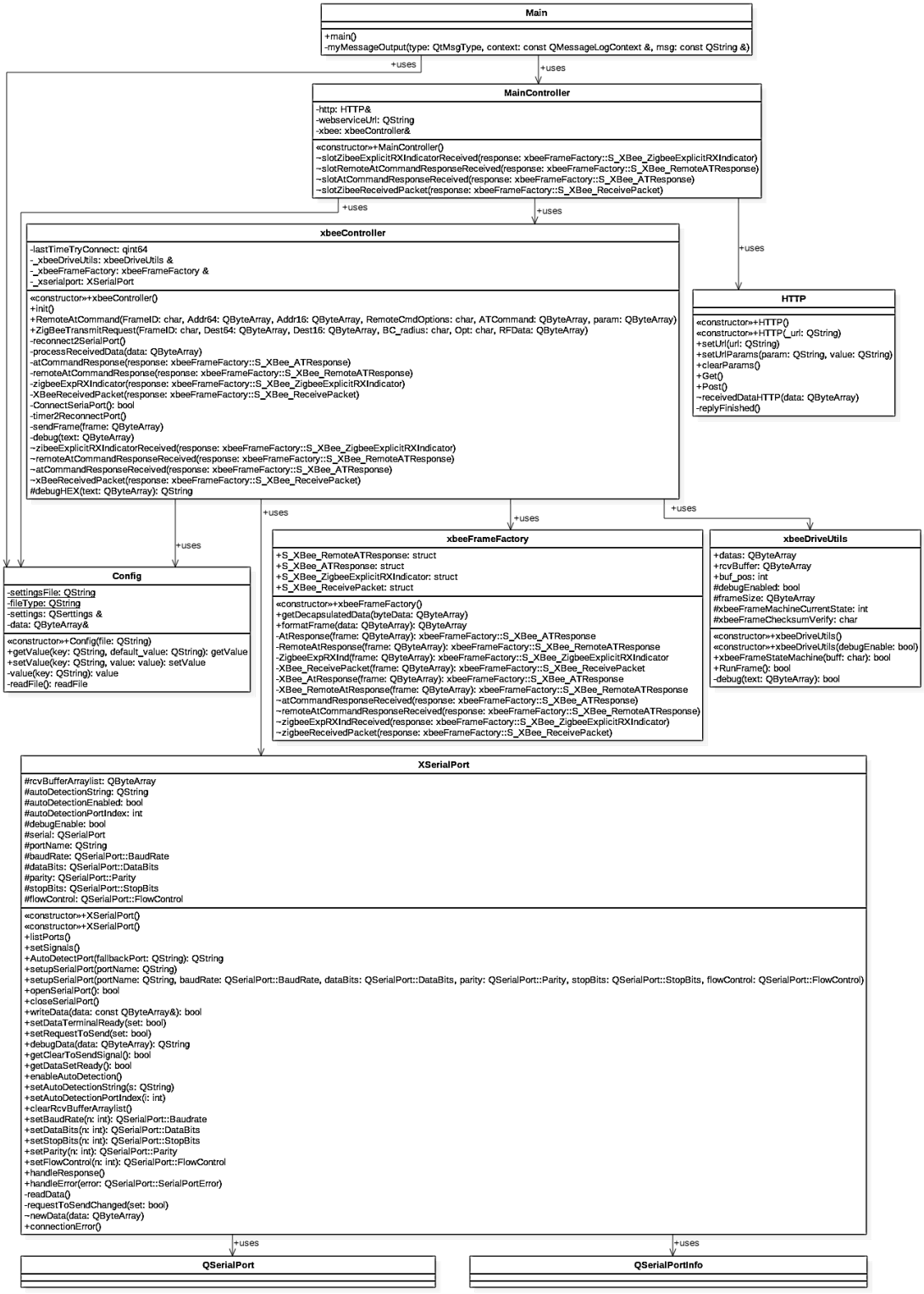


Figura 6.10: Diagrama de Classes no Router.

conteúdo que se pretende enviar.

No que toca à classe *Config*, esta foi desenvolvida com o intuito de poder fazer a configuração específica de alguns parâmetros necessários ao funcionamento do software (tais como indicar o servidor ao qual deve fazer os pedidos, porta série na qual está ligado o XBee, ativar níveis de *debug*, etc.). Esta configuração é feita a partir de um ficheiro externo ao programa, não sendo necessário alteração do software, para que este se possa adaptar facilmente a diferentes ambientes.

A classe *xbeeController* é a classe que deve sempre (em qualquer projeto que seja utilizado XBee) ser utilizada para fazer a interface entre todo o baixo nível relacionado com o XBee e a camada de negócio do software. Ao instanciar a *xbeeController* é possível enviar tramas do XBee apenas utilizando a chamada a métodos e receber os sinais (e correspondente informação) das diferentes tramas recebidas que depois podem ser atribuídos a eventos na camada superior.

De maneira a poder fazer os pedidos e receber a resposta a partir do protocolo HTTP foi criada a classe *HTTP*. A partir desta é possível definir o endereço ao qual se pretende fazer o pedido, definir parâmetros e fazer os pedidos utilizando tanto o método Get como Post.

A classe *MainController* é camada de mais alto nível existente neste software, uma vez que é nesta que são recebidos todos os sinais de tramas XBee e são traduzidos em ações, consoante o tipo de trama que recebe.

Por fim a classe *Main* é utilizada para instanciar a *MainController* e fazer o direcionamento do output da aplicação para um ficheiro de log. Desta forma é possível analisar tanto situações que se passaram, perceber o seu comportamento aos estímulos externos, assim como entender em tempo real o seu funcionamento. Consoante o nível de *debug* ativado, é possível obter uma informação mais detalhada em certas partes do código.

6.2.3 Descrição da Solução

O código desenvolvido para o Router seguiu uma programação orientada a eventos. Neste tipo de paradigma de programação, o fluxo que o programa vai ter é orientado por interrupções externas (os eventos) que ao receber informação para ser processada, desencadeia ações de resposta consoante o evento.

Neste software, o único evento que vai despoletar todo o processamento é o de receção de informação na porta série. No entanto, um evento pode ter diferentes tipos de ações conforme a informação que é recebida no buffer, sendo que neste caso cada trama tem um indicador de qual é o tipo de ação que se espera que este faça conforme é possível observar na Figura 6.11. Depois de receber o evento de informação recebida na porta série esta é guardada, posteriormente o buffer vai ser consumido em busca de uma trama válida e sempre que isto acontece é desencapsulada de maneira a poder ser interpretada. Conforme o comando recebido, é executada a ação correspondente, sendo que neste caso significa fazer algum tipo de pedido de informação ao servidor.

6.3 Web Service

O *Web Service* é o software que faz o processamento de pedidos por HTTP feitos pelas partes do sistema que necessitam de interagir (consultar/inserir/alterar) com a base de dados, nomeadamente o monitor de infusão e a interface com o utilizador.

6.3.1 Análise de Requisitos

Para este tipo de serviço é essencial que este utilize uma linguagem que consiga lidar com pedidos HTTP. Para tal foi necessário desenvolver um software em PHP que fosse capaz de receber os pedidos, processar consoante

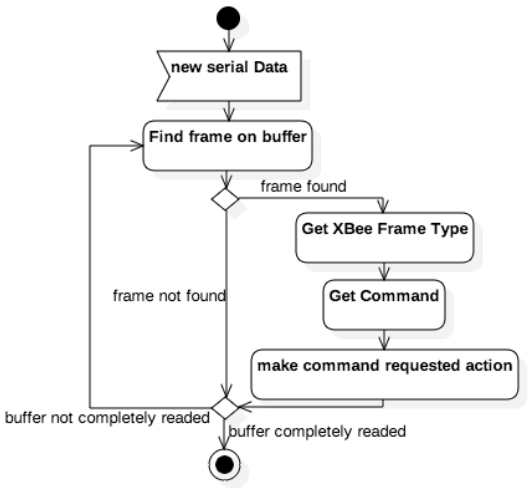


Figura 6.11: Diagrama de sequencia do processamento do evento de receção de informação na porta série.

o comando que nele estivesse presente e devolver o resultado da operação ao dispositivo que enviou o pedido.

Consoante o pedido recebido, este deve ser capaz de inserir registo de leitura, consultar/adicionar/alterar pacientes, consultar/adicionar/alterar medicamentos, consultar/adicionar/alterar dispositivos, consultar/adicionar/alterar atribuições de dispositivos a pacientes com a devida medicação e consultar/adicionar utilizadores.

6.3.2 Modulação do Código

Para a elaboração do *Web Service* estruturou-se o código da maneira que é possível observar no diagrama de classes da Figura 6.12. Desta forma foi possível fazer uma organização onde a receção dos pedidos, a interação com a base de dados e os dados necessários para o acesso à base de dados estão divididos em diferentes partes.

No que toca à funcionalidade de cada um dos blocos implementados, começando pelo nível mais baixo, temos o *database config*. Este bloco tem

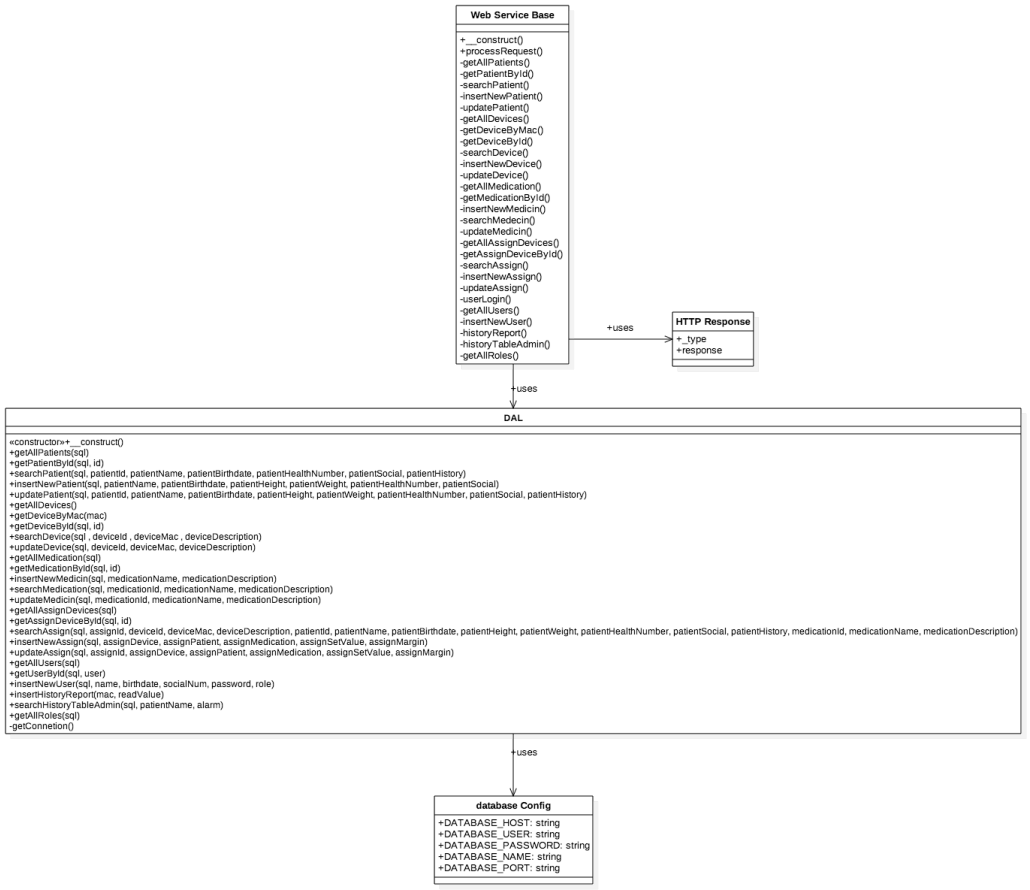


Figura 6.12: Diagrama de Classes do *Web Service*

como simples objetivo facultar os parâmetros necessários para que o nível superior saiba qual a base de dados que deve aceder e quais são as credenciais de acesso.

A classe DAL é a classe que se encarrega de preparar os parâmetros de entrada e a *query* de SQL (de maneira a não haver problemas de *SQL injection*), faz o pedido à base de dados e retorna o resultado para o nível superior.

A classe *Web Service Base* é a responsável por fazer a receção dos pedidos HTTP. Nesta o pedido é processado e é selecionada a ação correspondente ao tipo de pedido que é recebido.

Por fim classe *HTTP Response* é utilizada para uniformizar o tipo de resposta ao pedido enviado. Durante o envio do resultado da operação, pode assim ser criado um objeto que é preenchido com o tipo e a resposta obtida e posteriormente é convertida em JSON, para ser enviada e facilmente manipulada no destino.

6.3.3 Descrição da Solução

O *Web Service* é naturalmente feito baseado numa programação orientada a eventos, sendo que o evento que despoleta todas as ações é o recebimento de um pedido HTTP.

A partir do momento em que é recebido o pedido, este é primeiramente processado pelo *Web Service Base* que vai verificar o conteúdo do parâmetro "cmd"(que indica o tipo de processamento que deve ser feito), posteriormente é chamada a função que sabe como processar a informação contida no pedido (verificar campos obrigatórios) e depois chama o método da camada DAL que executa a *query* na base de dados para obter o resultado. O resultado retornado pela DAL é posteriormente encapsulado no *Web Service Base* num JSON (utilizando a classe HTTP Response) para ser enviado como resposta. De uma maneira mais geral é possível observar a sequencia

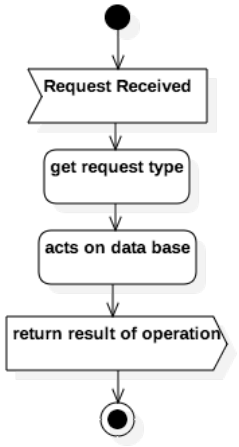


Figura 6.13: Diagrama de atividade do *Web Service*

de ações desde o momento em que é recebido o pedido HTTP no diagrama de atividade presente na Figura 6.13.

Código 6.5: Exemplo de processamento de pedido HTTP.

```
<?php
require_once('DAL.php') ;
$webservice = new WebService() ;
$webservice->processRequest() ;
class WebService
{
   ...
   const SEARCH_ASSIGN = "searchassign" ;
   ...
   public function processRequest ()
   {
     if ( !isset( $_REQUEST['cmd'] ) )
     { //if is not set
       echo json_encode( "Command is required for request
           processement" ) ;
       die() ;
```

```
    }
    $this->cmd = $_REQUEST['cmd'] ;
    switch ( $this->cmd )
    {
    ...
    case self::SEARCH_ASSIGN:
    self::searchAssign() ;
    break ;
    ...
    }
}
...
private function searchAssign ()
{
    $response = new HttpResponse() ;
    $response->_type = "assignSearch" ;
    $response->response = "nothing to search" ;
    $haveParametersToUpdate = false ;
    $sql = "SELECT assign_devices.`id` , devices.`id` AS d_id ,
        devices.`mac` AS d_mac , devices.`description`AS
        d_description , patients.`id` as p_id , patients.`name` as
        p_name , patients.`birthdate` as p_birthdate ,
        patients.`height` as p_height , patients.`weight` as
        p_weight , patients.`healthNum` as p_healthNum ,
        patients.`socialNum` as p_socialNum , patients.`history` as
        p_history , medication.`id` as m_id , medication.`name` as
        m_name , medication.`description` as m_description ,
        set_value , margin
    FROM `assign_devices`
    INNER JOIN patients ON patients.id = assign_devices.patient_id
    INNER JOIN devices ON devices.id = assign_devices.device_id
    INNER JOIN medication ON medication.id =
        assign_devices.medication" ;
```

```
if ( isset( $_REQUEST['id'] ) )
{
  $haveParametersToUpdate = true ;
  $sql .=" WHERE " ;
  $sql .= " assign_devices.`id` LIKE :id" ;
  $this->assignId = $_REQUEST['id'] ;
}
if ( isset( $_REQUEST['did'] ) )
{
  if ( $haveParametersToUpdate == true )
  {
    $sql .= " AND " ;
  }
  else
  {
    $haveParametersToUpdate = true ;
    $sql .=" WHERE " ;
  }
  $sql .= " devices.`id` LIKE :did" ;
  $this->deviceId = $_REQUEST['did'] ;
}
...
if ( $haveParametersToUpdate == true )
{
  $response->response = $this->dal->searchAssign( $sql ,
      $this->assignId, $this->deviceId , $this->deviceMac ,
      $this->deviceDescription , $this->patientId ,
      $this->patientName , $this->patientBirthdate ,
      $this->patientHeight , $this->patientWeight ,
      $this->patientHealthNumber , $this->patientSocial ,
      $this->patientHistory , $this->medicationId ,
      $this->medicationName , $this->medicationDescription ) ;
}
```

```
      echo json_encode( $response ) ;
   }
   ...
}
...
class HttpResponse
{
  public $_type = null ;
  public $response = null ;
  public function __construct() {}
}
...
<?php
require_once ('databaseConfig.php') ;
class DAL
{
  protected $connection ;
  public function __construct ()
  {
    $this->connection = $this->getConnection() ;
  }
  ...
  public function searchAssign ( $sql , $assignId, $deviceId ,
      $deviceMac , $deviceDescription , $patientId , $patientName ,
      $patientBirthdate , $patientHeight , $patientWeight ,
      $patientHealthNumber , $patientSocial , $patientHistory ,
      $medicationId , $medicationName , $medicationDescription)
  {
    try
    {
      $table = $this->connection->prepare( $sql ) ;
      if ( isset( $assignId ) && !is_null( $assignId ) )
      {
```

```
      $table->bindValue( ':id' , $assignId , PDO::PARAM_STR ) ;
    }
    if ( isset( $deviceId ) && !is_null( $deviceId ) )
    {
      $table->bindValue( ':did' , $deviceId , PDO::PARAM_STR ) ;
    }
    ...
    $table->execute() ;
    return $table->fetchAll( PDO::FETCH_ASSOC ) ;
  }
  catch ( PDOException $e )
  {
    return "false" ; //echo $e->getMessage();
  }
}
...
private function getConnection ()
{
  try
  {
    $newDatabaseConnection = new PDO( 'mysql:host=' .
        DATABASE_HOST . ';dbname=' . DATABASE_NAME .
        ';charset=utf8mb4' , DATABASE_USER , DATABASE_PASSWORD ) ;
    $newDatabaseConnection->setAttribute( PDO::ATTR_ERRMODE ,
        PDO::ERRMODE_EXCEPTION ) ;
    $newDatabaseConnection->setAttribute(
        PDO::ATTR_EMULATE_PREPARES , false ) ;
    return $newDatabaseConnection ;
  }
  catch ( PDOException $e )
  {
    echo $e->getMessage() ;
  }
```

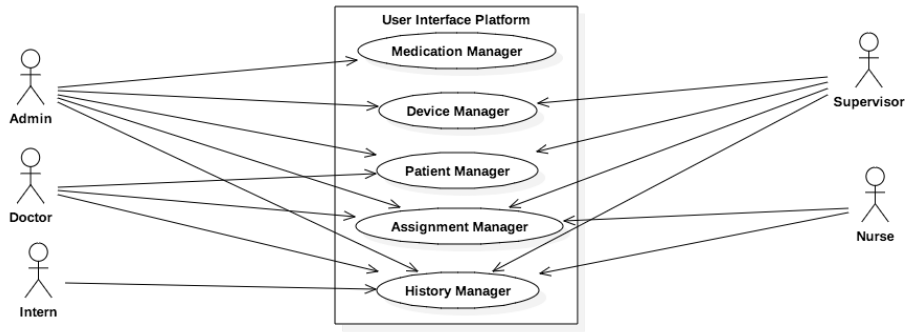


Figura 6.14: Diagrama de caso de uso do Interface com o Utilizador.

```
  }
}
```

6.4 Interface com Utilizador

A *interface com o utilizador* é um software que acaba por servir de plataforma para que um utilizador seja capaz de intuitivamente executar operações na base de dados.

6.4.1 Análise de Requisitos

Para o desenvolvimento da Interface com o utilizador pretende-se que o utilizador seja capaz de executar operações básicas de consulta, inserção e alteração de elementos da base de dados, sem que para tal seja necessário conhecer a arquitetura interna do sistema nem ter conhecimento técnico para executar as operações.

Como plataforma, apenas deve dar acesso ao utilizador conforme o seu papel havendo, portanto, restrições nas suas ações conforme as suas responsabilidades. Como tal o conjunto de ações que é permitido ao utilizador está representado na Figura 6.14.

Assim como é possível verificar na Figura 6.14 as ações foram agrupadas

em cinco diferentes grupos:

- *Medication Manager* - permite fazer a gestão da medicação (adicionar, consulta e alterar);
- *Device Manager* - permite fazer a gestão dos dispositivos (adicionar e alterar);
- *Patient Manager* - permite fazer a gestão dos pacientes (adicionar, consulta e alterar);
- *Assignment Manager* - permite fazer a gestão das atribuições paciente/dispositivo/medicamento (adicionar e alterar);
- *History Manager* - permite fazer a consulta do histórico dos valores de medicação registados ao longo do tempo e verificar valores anómalos.

6.4.2 Modulação do Código

No desenvolvimento da interface com o utilizador pode-se dividir o tipo de ações executadas nas seguintes partes:

- *Login*
- *Adição*
- *Consulta*
- *Alteração*

O processo de *Login* é o passo inicial ao abrir a aplicação. Neste pretende-se validar o utilizador para que seja possível pôr em prática todas as restrições de acesso às funcionalidades. Para que seja possível validar este passo é necessário desencadear o processo descrito na Figura 6.15. Caso a validação seja feita com sucesso irá ser apresentada em primeiro lugar a

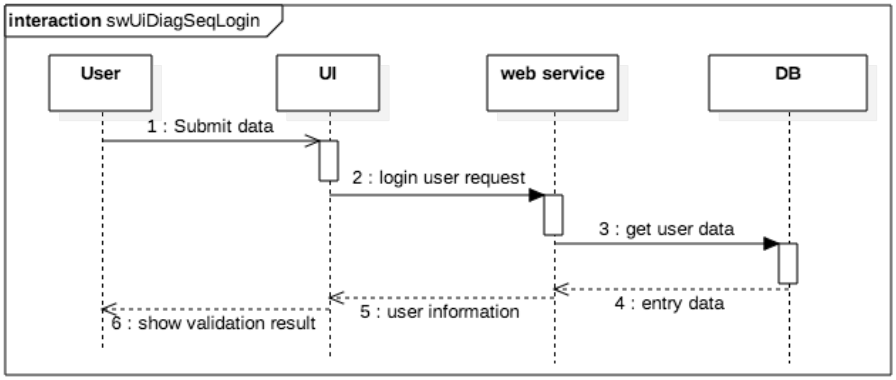


Figura 6.15: Diagrama de sequência para Login de utilizador.

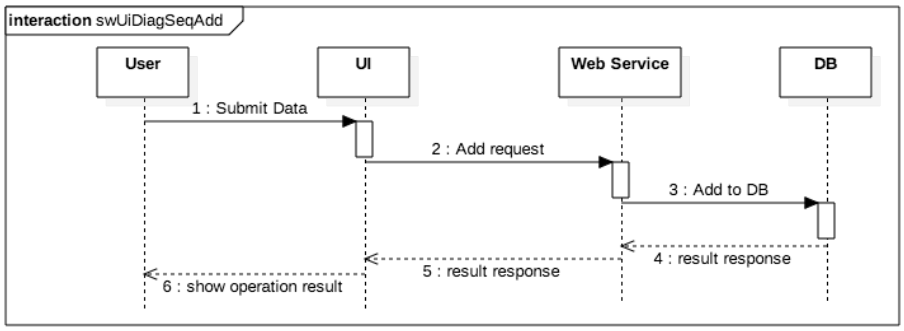


Figura 6.16: Diagrama de sequência para adição de registo

página de consulta de histórico dos pacientes, caso contrário será retornada uma mensagem de erro de autenticação e o utilizador poderá voltar a tentar.

O processo de *Adição* pretende fazer uma inserção de um registo na base de dados. Esta adição pode incidir na adição de um novo dispositivo, medicamento, paciente ou atribuição. Ao fazer o pedido para adicionar um registo, o fluxo de informação está representado na Figura 6.16. Como resultado desta operação é recebido uma mensagem de erro ou sucesso.

O processo de *Consulta* tem como objetivo apenas visualizar a informação de elementos tais como medicamentos, pacientes ou histórico. Esta consulta pode não só se aplicar a um elemento como a vários sendo, portanto, re-

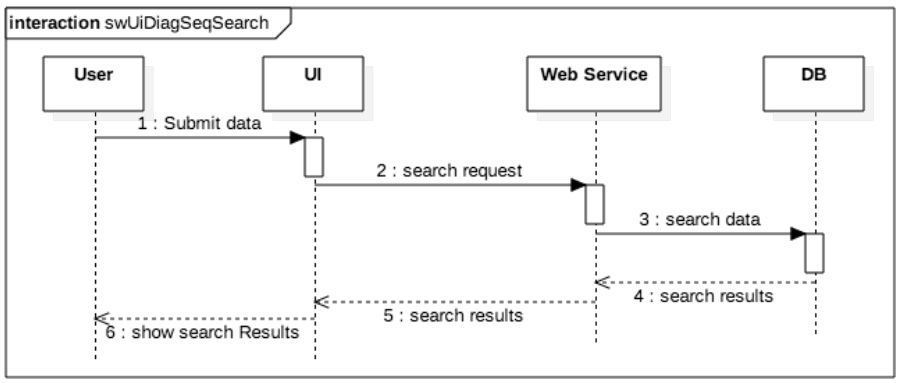


Figura 6.17: Diagrama de sequência para Consulta de dados.

tornada como resposta um conjunto de elementos (pode ser um ou mais) conforme os dados facultados para a pesquisa. Ao fazer o pedido para adicionar um registo, o fluxo de informação está representado na Figura 6.17. Como resultado desta operação é recebido uma mensagem com um vetor de elementos (caso venha vazio significa que não existe correspondência para aquela pesquisa).

De maneira a poder desenvolver o software, a classe responsável pela manipulação da interface gráfica teve de fazer uso de diferentes classes de suporte para lidar com protocolos, tipos de elementos e permitir um certo nível de configurações básico. Como resultado da operação de codificação deste software resultou o diagrama de classes representado na Figura 6.18.

6.4.3 Descrição da Solução

Nesta solução o código desenvolvido é orientado a eventos. Existem duas fontes de eventos nomeadamente os desencadeados pelo utilizador ao interagir com a interface gráfica ou pela reposta a um pedido HTTP ao *Web Service*. No caso dos eventos desencadeados pelo utilizador (botões pressionados, duplo clique em tabelas, ...) pode existir antes da ação pretendida uma validação dos dados preenchidos (no caso de submissão de pedidos de

Figura 6.18: Diagrama de Classes da Interface com Utilizador

adição e alteração de registos da base de dados), podendo esta validação ser feita pela verificação de preenchimento de campos obrigatórios ou formatação inválida dos dados (Figura 6.19a).

Aquando de um duplo clique em determinadas tabelas (consulta de histórico e tabelas de consulta) este pode ter dois tipos de ação, nomeadamente a abertura de uma nova janela com mais detalhes sobre a informação selecionada (Figura 6.19b) ou preenchimento dos campos baseados no registo selecionado (Figura 6.19c). Este último caso é mais utilizado quando se pretende proceder à alteração do registo.

Noutra vertente existem os eventos desencadeados por resposta aos pedidos por protocolo HTTP. Neste caso estas vêm formatadas em JSON e tem na sua constituição dois campos diferentes: "cmd"e "response". No campo "cmd"está indicado qual é o tipo de informação que é essencial para saber qual o tipo de ação que tem de ser levada a cabo, o campo "response"vem preenchido com o conteúdo da mensagem que pode ser só uma variável de sucesso ou erro ou um vetor de entradas de registos na base de dados caso seja um pedido de informação de registos. O tratamento dos dados contidos no campo de "response"vai sempre depender do tipo de comando especificado no campo "cmd".

Uma vez especificada toda a arquitetura utilizada no desenvolvimento dos softwares apresentados em cima, podemos concluir que a parte prática do projeto está finalizada. Uma vez terminada a fase de desenvolvimento do projeto, iremos passar para uma fase onde se pretende mostrar no próximo capitulo quais foram os testes efetuados ao sistema e os resultados obtidos pelo mesmo.

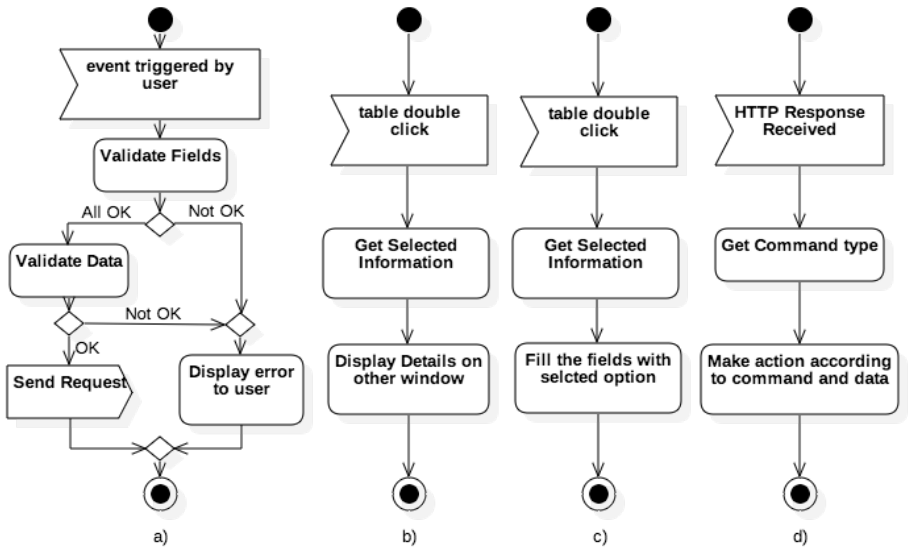


Figura 6.19: Diagrama de atividade do processamento dos diferentes eventos da interface com o utilizador: a) despoletado pelo utilizador; b) duplo clique numa tabela que abre detalhes; c) duplo clique numa tabela que preenche campos; d) receção de resposta HTTP.

Capítulo 7

Testes e Resultados

Este capítulo destina-se a apresentar o desenvolvimento do protótipo do monitor de infusão, expor os testes aos quais o sistema foi submetido para verificar o seu correto funcionamento e quais foram os resultados obtidos no final de todos os testes.

7.1 Desenvolvimento de protótipo do Monitor de Infusão

Uma vez idealizado e totalmente dimensionado todo o circuito para o desenvolvimento do Monitor de Infusão, foi necessário avançar para a construção do protótipo do mesmo. Nesta fase do desenvolvimento foi utilizada:

- Placa de desenvolvimento STM32L100C-DISCO;
- Módulo XBee;
- Conversor de pinout de XBee (2mm) para placa perfurada de (2,54mm);
- Módulo de alimentação com TP4056;
- Bateria de lítio de 1000mAh;
- Placa perfurada de 2,54mm de espaçamento;

Na fase de prototipagem foi utilizada a placa de desenvolvimento STM32L100C-DISCO, no sentido de facilitar a montagem e programação do microcontrolador. Embora em funcionamento só seja utilizada uma pequena parte da placa, esta acaba (para o tipo de dispositivo que se pretende desenvolver) por ocupar bastante espaço, tendo sido dividido todo o hardware em duas partes:

1. Sensor (Figura 7.1a e 7.1b):
    - Detetor de Gotas;
    - Condicionamento de Sinal.
2. Processamento(Figura 7.1c):
    - Alimentação (módulo com TP4056);
    - Microcontrolador (placa de desenvolvimento);
    - Módulo XBee (Comunicação).

As duas partes em que o hardware foi separado, são conectadas a partir de um cabo onde são passados: a alimentação (VCC e Ground) e o sinal condicionado. Após a conexão as duas partes do hardware, estas têm exatamente o mesmo desenho e comportamento esperado do circuito desenvolvido. Como resultado final do desenvolvimento do protótipo do Monitor de Infusão, resultou o dispositivo representado na Figura 7.2.

7.2 Ensaios com o Detetor de Medicação

De maneira a fazer testes ao monitor de infusão foram feitas algumas experiências para comprovar o seu correto funcionamento em diferentes condições. De modo a efetuar os testes, foi necessário alterar algumas condições externas ao sistema que se pensou que pudessem de alguma forma induzir em erro o sistema.

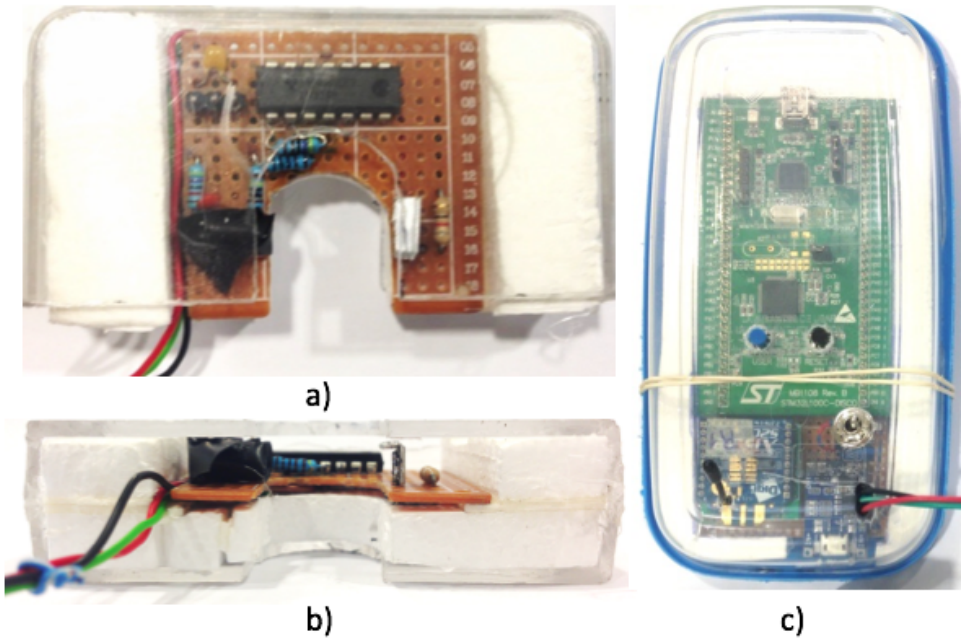


Figura 7.1: a) Vista superior da parte do Sensor do Monitor de Infusão; b) Vista frontal da parte do Sensor do Monitor de Infusão; c) Parte de hardware correspondente ao Processamento do Monitor de Infusão

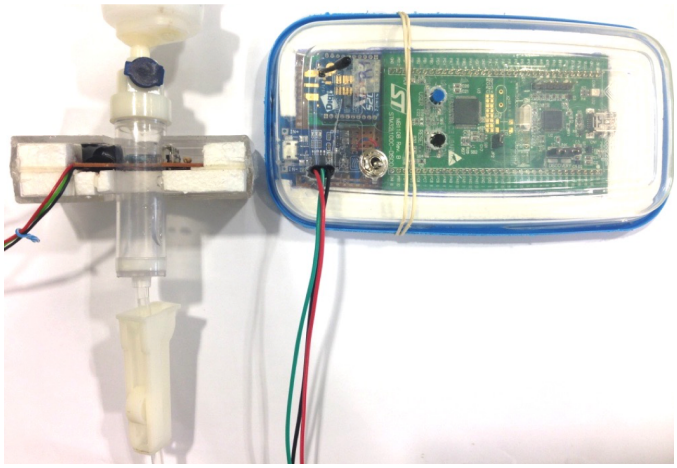


Figura 7.2: Protótipo Final do hardware do Monitor de Infusão

7.2.1 Teste de Luminosidade Ambiente

Um primeiro teste permitiu verificar que a luminosidade externa (tanto por luz artificial como em diferentes alturas do dia) não tinham nenhuma influência significativa no sistema, também devido ao isolamento aplicado no recetor para não receber radiação lateral, mas apenas frontal (onde se encontra o emissor).

7.2.2 Teste com Características do Fluido

Outro fator que foi tido em conta nos testes realizados foi a alteração das propriedades do líquido que poderia ser administrado. Neste sentido, foram efetuados testes com líquidos alternativos ao soro, onde se pretendeu testar o comportamento do sistema com fluidos onde características como a opacidade, cor e viscosidade eram alteradas.

Controlo

Numa fase inicial foi feito o teste de controlo utilizando o soro que foi o líquido base no desenvolvimento e que possui as características típicas de medicação administrada. Este fluido apresenta viscosidade baixa, cor transparente e nenhuma opacidade.

A resposta do circuito a este líquido está representada na Figura 7.3. Esta é a resposta esperada para este tipo de líquido, uma vez que é transparente e desta forma irá ocorrer refração da luz (resultando num duplo pico de tensão). Dado que este líquido apresenta uma baixa viscosidade o tamanho da gota será de menores dimensões (uma vez que a sua tensão à superfície é baixa), o que reflete o seu formato esférico e desta forma podemos observar que a resposta em tensão representa a refração da luz na gota ao longo da sua queda.

Tendo em conta que o microcontrolador, teoricamente, deteta o impulso a partir dos ≈1,89V (na prática verificou-se deteção acima de ≈1,5V) pode-

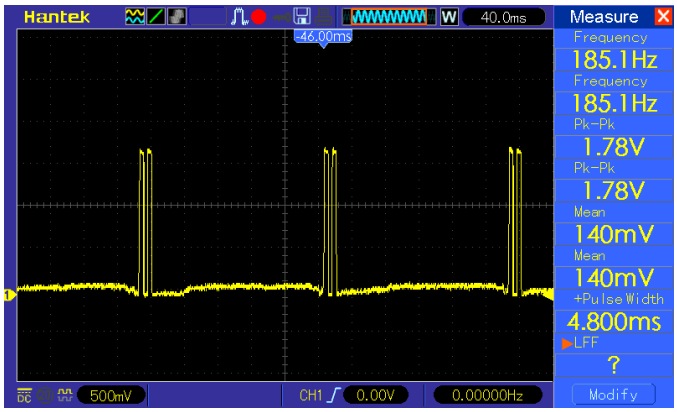


Figura 7.3: Teste do detetor de medicação com Soro Simples (controlo).

se concluir o valor de tensão detetado neste teste é perfeitamente aceitável para que o sistema consiga fazer a deteção nestas condições.

Alteração de Cor

Após ter sido efetuado o teste de controlo procedeu-se ao teste do soro com corante alimentar (xarope). Desta forma o soro que se apresentava como transparente e com viscosidade reduzida passou a tomar características diferentes, nomeadamente coloração acastanhada, pequeno aumento da opacidade e ligeira viscosidade. O resultado do teste pode ser verificado na Figura 7.4. Tendo em conta que os níveis de tensão requeridos por parte do microcontrolador, pode-se concluir que o sistema consegue fazer a deteção nestas condições.

Alteração de Viscosidade

Seguindo ainda os testes de viscosidade dos líquidos, foi feito posteriormente uma análise do comportamento do sistema com óleo alimentar. Este fluido apresenta uma baixa coloração, baixa opacidade, mas elevada viscosidade. Como resultado da experiência, resultou a resposta presente na Figura 7.5. Uma vez que este líquido apresenta maior viscosidade, a sua tensão à superfície é superior, o que resulta em gotas de maior dimensão. O facto de

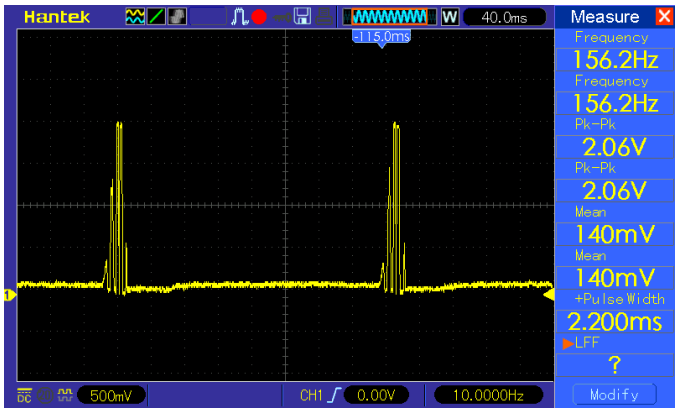


Figura 7.4: Teste do detetor de medicação com Soro e Corante Alimentar.

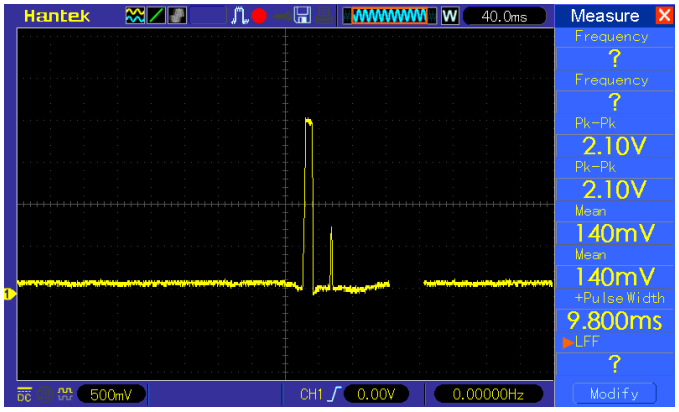


Figura 7.5: Teste do detetor de medicação com Óleo Alimentar.

serem formadas gotas de maiores, faz com que durante a queda esta se possa deformar e, desta forma, tomar uma forma diferente da esférica (como no caso do soro que apresenta viscosidade baixa). Esta alteração do formato faz com que a refração da luz na gota tenha uma resposta diferente e desta forma resultam diferentes picos durante a deteção. Apesar da resposta ser diferente dos fluidos menos viscosos, pode-se observar que o sistema continua a ser capaz de fazer a deteção da gota, uma vez que a sua resposta é ≈2V que é superior à tensão necessária para que haja interrupção no microcontrolador.

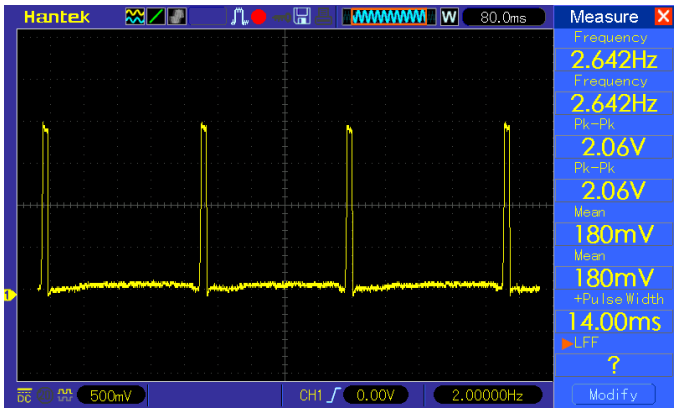


Figura 7.6: Teste do detetor de medicação com Leite.

Alteração de Opacidade

Como teste final de características dos líquidos, foi feito o teste com leite. Este líquido apresenta baixa viscosidade, no entanto apresenta uma elevada opacidade. A resposta do circuito, quando um fluido com características semelhantes às do leite passa pelo sistema, é representado na Figura 7.6. Uma vez que o líquido apresenta uma elevada opacidade, o feixe luminoso não é capaz de atravessar a gota, o que faz com que a resposta seja uma onda quadrada, uma vez que não há refração da luz. No entanto, apesar de a gota bloquear totalmente a passagem da luz e só existir um pico, pode-se concluir que os níveis de tensão são perfeitamente satisfatórios para a deteção da gota por parte do microcontrolador.

7.3 Base de Dados

A base de dados encontra-se implementada no mesmo servidor que o *Web Service*, e é um elemento de grande importância no sistema pois permite guardar os registos de valores lidos, lista de pacientes, lista de medicamentos, lista de dispositivos, lista de associações (entre dispositivos, paciente, medicação, valor esperado e margem), lista de utilizadores do sistema e cargos/permissões dos utilizadores.

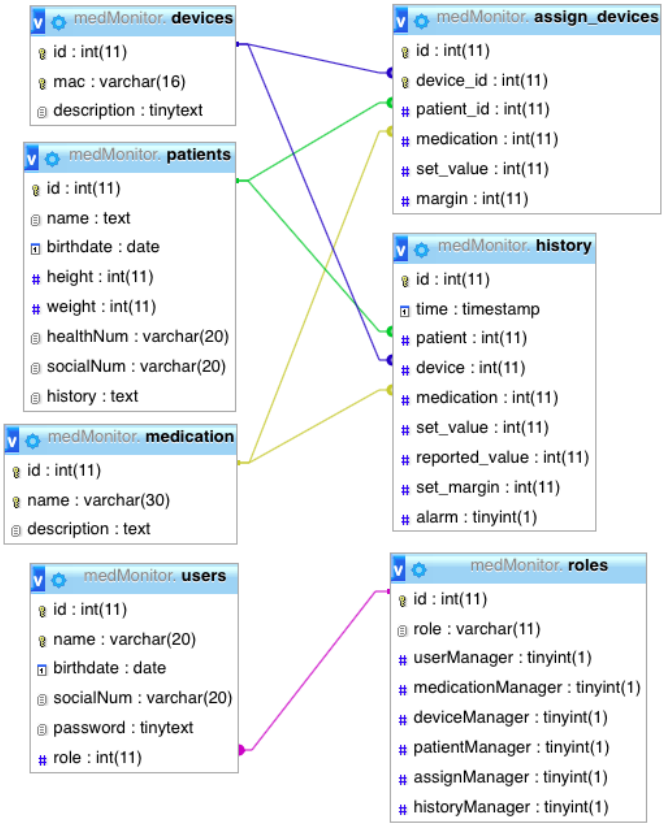


Figura 7.7: Estrutura final da base de dados relacional.

7.3.1 Estruturação da Base de Dados

No final do desenvolvimento a base de dados relacional ficou com a estrutura representada na Figura 7.7.

Nesta é possível verificar a existência de sete tabelas diferentes:

- *devices* - onde são registados todos os dispositivos que fazem leituras de medicação.
- *patients* - tabela que contém todos os dados dos pacientes;
- *medication* - contém todos os medicamentos existentes;
- *assign_devices* - tabela que faz a associação entre o dispositivo, o paciente, o medicamento, valor esperado e a margem (em percentagem)

aceitável para não ser despoletado o alarme;

- *history* - tabela que contém os registos dos valores lidos nos dispositivos. Cada registo possui informação do paciente, dispositivo, medicamento associado, valor lido, valor esperado na altura e um campo que indica valores fora do esperado.

- *users* - tabela onde são registados todos os utilizadores, para que estes tenham acesso à interface como utilizador.

- *roles* - contém os diferentes cargos/responsabilidades e indica que tipo de ações são permitidas. Estes cargos são atribuídos aos pacientes para que ao fazer login na plataforma haja restrições de acesso consoante o utilizador.

7.3.2 Registo de Leituras

Todos os valores provenientes dos Monitores de Infusão são, através do *Web Service*, introduzidos na tabela *history*. De modo a ser possível preencher todos os campos, no pedido ao *Web Service* é enviado o endereço MAC do dispositivo de maneira a poder descobrir o identificador na tabela *devices*, pesquisar qual a entrada na tabela de associação (*assign_devices*) e de seguida aproveitar os identificadores únicos dos diferentes parâmetros desta para preencher a tabela *history* (onde são guardados os registos). Cada entrada possui os valores de medicação prescritos, campo de alarme (para valores anómalos aos esperados), identificadores únicos dos pacientes, medicamentos e dispositivo, tal como é possível verificar na Figura 7.8. Os identificadores de paciente, medicamentos e dispositivos estão diretamente associados às respetivas tabelas que contém os dados atuais por*foreign key* do SQL.

	id	time	patient	device	medication	set_value	reported_value	set_margin	alarm
Edita Copiar Apagar	65	2017-03-04 00:18:58	29	1	2	30	33	20	0
Edita Copiar Apagar	63	2017-03-04 00:14:58	29	1	2	30	24	20	0
Edita Copiar Apagar	62	2017-03-04 00:13:57	29	1	2	30	54	20	1
Edita Copiar Apagar	61	2017-03-04 00:11:57	29	1	2	30	41	20	1
Edita Copiar Apagar	60	2017-03-04 00:09:58	29	1	2	30	42	20	1
Edita Copiar Apagar	59	2017-03-04 00:07:57	29	1	2	30	58	20	1
Edita Copiar Apagar	57	2017-03-04 00:03:58	29	1	2	30	24	20	0
Edita Copiar Apagar	56	2017-03-04 00:02:57	29	1	2	30	14	20	1
Edita Copiar Apagar	55	2017-03-04 00:00:57	29	1	2	30	8	20	1
Edita Copiar Apagar	54	2017-03-03 23:58:57	29	1	2	30	8	20	1

Figura 7.8: Registos de Leituras de medicação na base de dados, tabela *history*.

7.4 Interface com Utilizador

A interface do utilizador surge como uma plataforma de *desktop*, para utilizadores registados poderem gerir algumas informações da base de dados. Inicialmente a interface apresenta uma página de Login (Figura 7.9) para que o utilizador possa executar alguma ação na plataforma.

Uma vez que o utilizador valide as suas credenciais aparece a primeira página que mostra os registos de valores reportados com a opção de *"Only Alarmed"* ativa. Na Figura 7.10 apresenta-se a visualização do histórico de pacientes com a pesquisa feita pelo nome e sem a seleção de *"Only Alarmed"* ativa (de maneira a ver todos os resultados), onde se podem verificar os valores dentro dos parâmetros a verde e os anómalos a vermelho.

Caso durante a consulta do histórico dos pacientes os utilizadores pretenderem obter mais informações sobre o paciente ou medicação, basta fazer duplo-clique sobre a célula da tabela para obter informações mais detalhadas sobre o elemento em questão, conforme é ilustrado na Figura 7.11.

Figura 7.9: Login da Interface com Utilizador.

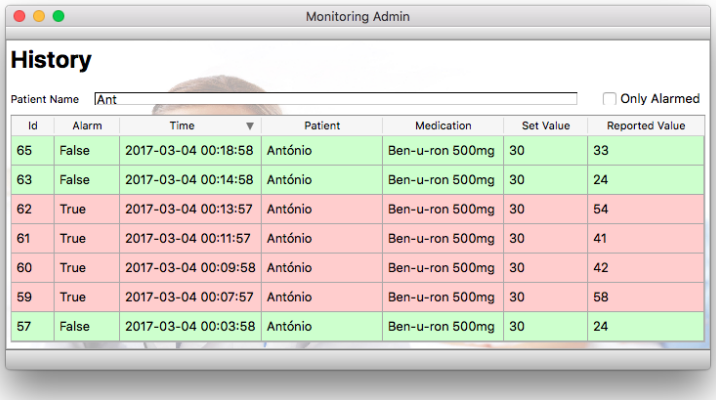


Figura 7.10: Visualizador de Histórico de Registos da Interface com Utilizador.

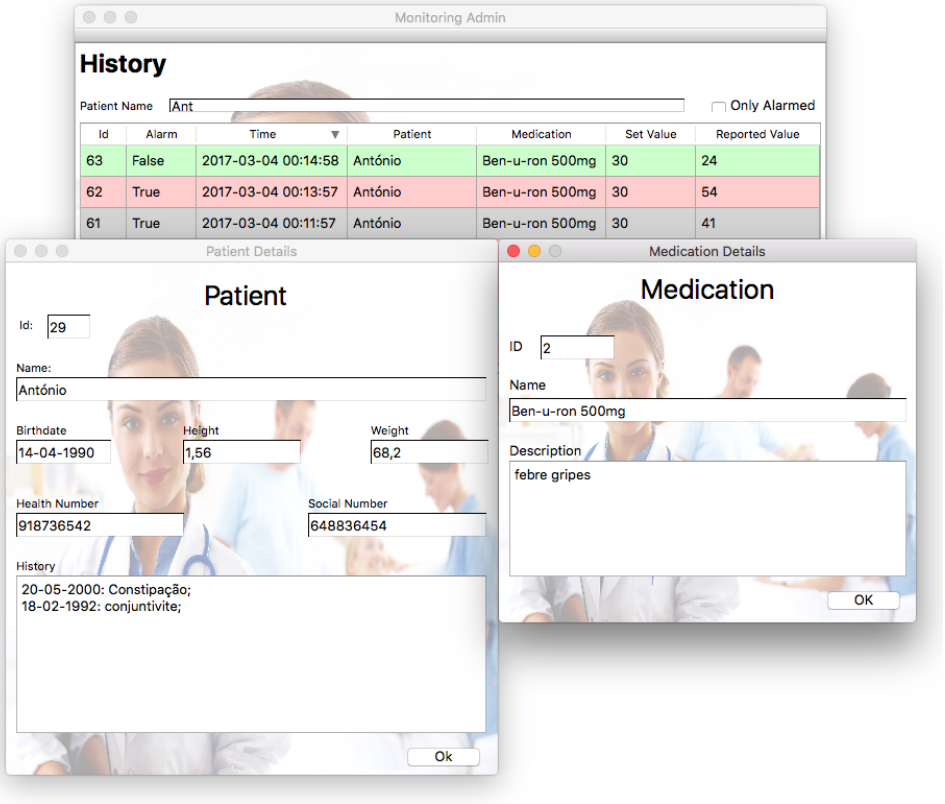


Figura 7.11: Detalhes sobre Paciente e Medicação no Histórico da Interface com Utilizador.

Capítulo 8

Conclusão e Trabalho Futuro

Ao longo deste projeto foram sendo apresentadas algumas conclusões que permitiram dar suporte às opções de desenvolvimento feitas. Desta forma pretende-se nesta ultima secção realizar um resumo das conclusões mais importantes e perspetivar alguns futuros desenvolvimentos.

Numa primeira fase foi feita uma análise dos principais dispositivos já desenvolvidos nesta área, esta pesquisa permitiu entender não só que tipos de dispositivos são os que o mercado procura, e assim modular de alguma forma o sistema para corresponder às necessidades, como também perceber algumas técnicas que são utilizadas para a deteção da medicação de uma forma comprovadamente fiável. Este estudo permitiu de igual forma auxiliar na decisão de qual seria a técnica utilizada para a deteção da medicação.

Posteriormente foi feita a implementação do protocolo wireless ZigBee para a comunicação entre o monitor de infusão e o Router. Este protocolo foi escolhido, no meio de outros encontrados, por ser de baixo consumo energético e ser certificado para aplicações médicas e por utilizar a tecnologia *mesh* que permite uma fácil implementação quando se pretende falar entre quaisquer dois pontos na rede.

Uma vez ultrapassada a fase de pesquisa, durante o desenvolvimento, foi possível desenvolver com sucesso: o módulo de deteção de medicação

que permitisse a mobilidade, um elemento que submetesse as leituras num sistema centralizado, um serviço que fosse capaz de responder a pedidos externos e fosse capaz de interagir com a base de dados e por fim uma plataforma para gerir os registos da base de dados.

Como perspetivas futuras de trabalho inicialmente seria necessário fazer ensaios clínicos e obter certificação médica obedecendo a normas (ISO 13485) ou haver alguém interessado que já tivesse adquirido as normas para normalizar e implementar o sistema. Posteriormente seria necessário um trabalho adicional para que fosse possível a integração deste sistema com a base de dados ou um serviço já implementado numa instituição de saúde, para que este pudesse ter um caráter universal e ser implementado em diferentes locais.

Bibliografia

[1] A. A. Ramos, "Seminário - dispositivos de infusão." [Online]. Available: https://pt.slideshare.net/andalbram/seminrio-dispositivos-de-infuso

[2] P. S. Ganney, S. Pisharody, and E. McDonagh, *Clinical Engineering - PRINCIPLES AND APPLICATIONS IN ENGINEERING SERIES*, 2005. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B978012396961300010X

[3] "Infusion equipment and intravenous anaesthesia — Clinical Gate." [Online]. Available: http://clinicalgate.com/infusion-equipment-and-intravenous-anaesthesia/

[4] M. P. Agency, "Report Infusion pumps," vol. 46, no. August, 2014.

[5] S. Range, C. Network, E.-r. Digimesh, E.-r. Zigbee, F. Factor, D. Kit, P. Numbers, C. Regions, and S. Mount, "Digi XBee ® Family Features Comparison Small , Simple , Pre-certified and Ready to Communicate," 2017.

[6] Digi International, "Wireless Connectivity Kit Getting Started Guide Getting Started Guide," 2016.

[7] ——, "XBee/XBee-PRO S2C 802.15.4 Radio Frequency (RF) Module," 2013.

[8] Ministério da Saúde, "DL nº 176/2006 de 30 de agosto," *Diário da República 1ª série*, pp. 6297–6303, 2006. [Online]. Available: http://www.portaldasaude.pt/NR/rdonlyres/D2D959FC-A937-4850-B0DC-B60E04F2108B/0/62976383.pdf

[9] Royal College of Nursing, "Standards for infusion therapy," *R. Coll. Nurs.*, 2010. [Online]. Available: http://www.rcn.org.uk/{_}{_}data/assets/pdf{_}file/0005/78593/002179.pdf

[10] M. D. M. Lourenço, "Sistema sem fios para controlo de iluminação decorativa," Ph.D. dissertation, Instituto Superior de Engenharia do Porto, 2011.

[11] Octavian, "WPAN – ZigBee & Bluetooth," pp. 1–5, 2014.

[12] J.-S. Lee, Y.-W. Su, and C.-C. Shen, "A Comparative Study of Wireless Protocols: Bluetooth, UWB, ZigBee, and Wi-Fi," *33rd Annu. Conf. IEEE Ind. Electron. Soc. (IECON 2007)*, pp. 46–51, 2007.

[13] N. Baker, "ZigbeeVsBluetooth," *Comput. Control Eng. J.*, vol. 16, no. 2, pp. 20–25, 2005.

[14] J.-c. Chou, J.-t. Chen, Y.-h. Liao, and C.-h. Lai, "Wireless Sensing System for Flexible Arrayed Potentiometric Sensor Based on XBee Module," vol. 16, no. 14, pp. 5588–5595, 2016.

[15] S. Kasapovic, E. Doric, and L. Banjanovic-mehmedovic, "An Approach to Wireless Sensor Network Design in Home Environment using ZigBee Protocol," *2015 23rd Int. Conf. Software, Telecommun. Comput. Networks*, 2015.

[16] ZigBee Alliance, "ZigBee Certified — The ZigBee Alliance." [Online]. Available: http://www.zigbee.org/zigbee-for-developers/zigbeecertified/

[17] M. Mushaib, R. Paulus, A. K. Jaiswal, and A. Agarwal, "Analyzing and Shrinking Network Inaccessibility in ZIGBEE Wireless Communication by Comparing it with TDMA-MAC Protocol," *2016 5th Int. Conf. Reliab. Infocom Technol. Optim. (Trends Futur. Dir.*, pp. 7–12, 2016.

[18] M. Z. Residential, S. Protocol, and D. P. Lattibeaudiere, "AN1232, Microchip ZigBee-2006 Residential Stack Protocol," 2008.

[19] Network Working Group, "RFC 2616 - Hypertext Transfer Protocol (HTTP/1.1)," 1999. [Online]. Available: http://www.rfc-base.org/txt/rfc-2616.txt

[20] Internet Engineering Task Force, "RFC 6455 - The WebSocket Protocol," 2011. [Online]. Available: http://www.rfc-editor.org/info/rfc6455.

[21] Digi International, "XBee buying guide - Wireless Connectivity Kit - Digi Docs." [Online]. Available: https://docs.digi.com/display/WirelessConnectivityKit/XBee+buying+guide

[22] ST Microeletronics, "STM32L series Ultra low power 32bit MCUs Releasing your creativity." [Online]. Available: http://www.st.com/content/ccc/resource/sales{_}and{_}marketing/promotional{_}material/brochure/6c/48/c0/f1/bb/35/4a/b4/brstm32ulp.pdf/files/brstm32ulp.pdf/jcr:content/translations/en.brstm32ulp.pdf

[23] "A Brief Description - C++ Information." [Online]. Available: http://www.cplusplus.com/info/description/

[24] "History of C++ - C++ Information." [Online]. Available: http://www.cplusplus.com/info/history/

[25] "PHP: Hypertext Preprocessor." [Online]. Available: http://php.net/

[26] "Keil Embedded Development Tools for ARM, Cortex-M, Cortex-R4, 8051, C166, and 251 processor families." [Online]. Available: http://www.keil.com/

[27] "STM32CubeMX - STM32Cube initialization code generator - STMicroelectronics - STMicroelectronics." [Online]. Available: http://www.st.com/en/development-tools/stm32cubemx.html

[28] Qt Company, "The Qt Company - Ensuring the future is written with Qt." [Online]. Available: https://www.qt.io/company/

[29] Texas Instruments, "TPS61090 2A Switch, 96% Efficient Boost Converter — TI.com." [Online]. Available: http://www.ti.com/product/tps61090

[30] Microchip, "MCP73833 - Battery Management and Fuel Gauges - Battery Management and Fuel Gauges - Battery Chargers." [Online]. Available: http://www.microchip.com/wwwproducts/en/en027785

[31] NanJing Top Power ASIC Corp., "TP4056 1A Standalone Linear Li-Ion Battery Charger with Thermal Regulation in SOP-8." [Online]. Available: https://dlnmh9ip6v2uc.cloudfront.net/datasheets/Prototyping/TP4056.pdf

[32] ST Microeletronics, "STM32L100RC microprocessor Datasheet," no. March, 2015.

[33] T. Instruments, "LMx24-N , LM2902-N Low-Power , Quad-Operational Amplifiers," *Datasheet*, vol. SNOSC16D, 2015.