UNIVERSIDADE DE COIMBRA

Intellicare

Sistema de receção, armazenamento e visualização de dados de Oximetria e ECG

Gonçalo Dias Coimbra Vigário Louzada

Mestrado em Engenharia Biomédica

Faculdade de Ciências e Tecnologias – Departamento de Física

Setembro 2013

Sistema de receção, armazenamento e visualização de dados de Oximetria e ECG

Departamento de Física

Faculdade de Ciências e Tecnologias

Universidade de Coimbra

Orientadores

Eng. Andreia Carreiro

Eng. António Damasceno

Prof. Dr. Carlos Correia

Prof. Dr. José Basílio Simões

Gonçalo Dias Coimbra Vigário Louzada

Agradecimentos

Não poderia finalizar este longo percurso sem agradecer àqueles que contribuíram inequivocamente para que conseguisse desenvolver todo o trabalho. Sem as pessoas e instituição que me acompanharam, tudo seria mais difícil.

Começo por agradecer à ISA – *Intelligent Sensing Anywhere, S.A.* e seus colaboradores por toda a ajuda disponibilizada, muitas vezes prescindindo de tempo precioso do seu trabalho para me poderem ajudar. O ambiente e condições providenciadas foram uma grande contribuição para o sucesso deste projeto.

Um agradecimento especial à Engenheira Andreia Carreiro que me guiou ao longo deste extenso caminho, providenciando ajuda preciosa, especialmente nos momentos de maior desorientação. Um obrigado pelas várias e valiosas lições que tão importantes foram para que tudo corresse como desejado.

Também quero agradecer ao Engenheiro António Damasceno, que com os seus conhecimentos e experiência, me ajudou sempre a encontrar as soluções necessárias e a enquadrar as mesmas no conceito do projeto. A sua orientação técnica foi muito importante no desenvolver de todo o trabalho.

Quero agradecer ao Engenheiro Pedro Marques, que esteve sempre disponível para me ajudar com todas as questões técnicas inerentes a este projeto.

Aos meus amigos que tanto me ajudaram, nos bons e nos maus momentos, o meu obrigado por anos que nunca serão esquecidos, bem como nenhum de vocês.

Um obrigado a ti Daniel, que me acompanhaste nesta aventura e me soubeste sempre ajudar quando precisei. A tua amizade foi uma das forças que contribuíram para o sucesso deste desafio.

O projeto que concluo seria impossível sem o apoio da minha família. Aos meus pais, por estarem sempre do meu lado e me incentivarem na busca da realização pessoal, o meu eterno obrigado. Aos meus irmãos, por preencherem um espaço na minha vida que sem eles ficaria vazio, e por me ajudarem a relaxar nos tempos livres, o meu muito obrigado. Aos meus tios, tias, e aos meus avós, que ainda hoje fazem parte do meu crescimento, o meu obrigado. Ao *Néu*, que tanto me ensinou e partilhou comigo, obrigado.

A ti Susana, que durante este ano tanto apoio que me ofereceste, o meu obrigado. Obrigado por todos os momentos em que me mostraste que eu era capaz de cumprir aquilo a que me propunha. Por me mostrares a luz, mesmo quando eu não fazia ideia onde procurar. Por me fazeres acreditar. A tua dedicação e afeto incondicionais são uma força que espero poder retribuir.

Resumo

O envelhecimento da população, associado à melhoria da qualidade de vida e consequentemente a uma esperança média vida cada vez maior, acarreta consigo grandes desafios no que toca à manutenção dessa qualidade de vida com o aumentar da idade. As doenças tornam-se mais prováveis, e entre estas figuram as doenças do foro cardiovascular, que constituem das principais causas de morte mundiais.

Um produto que permita ao seu utilizador realizar um dia-a-dia normal, ao mesmo tempo que monitoriza parâmetros biomédicos para uma avaliação constante do seu estado de saúde, é assim uma resposta viável à necessidade de análise frequente do estado de saúde de um individuo. Este projeto visa desenvolver ferramentas a ser utilizadas num produto móvel de monitorização permanente de dados vitais já existente, o OneCare da Intellicare, uma *spin-off* da ISA – Intelligent Sensing Anywhere, S.A..

Foram utilizados vários sensores portáteis de sinais biomédicos, para obter parâmetros vitais que indicam se um individuo se encontra em estado normal, ou a sofrer de uma situação anormal. Os sensores enviam os seus dados através de Internet para um programa remoto, que recebe os dados e os armazena. Um segundo programa permite depois visualizar graficamente os dados recebidos. Este projeto focou-se no desenvolvimento dos programas responsáveis por receber, descodificar, armazenar e representar graficamente os dados biomédicos.

Palavras-chave: Java, MQTT, JSF, Armazenamento de Dados, Monitorização Remota de Sinais Vitais

Abstract

The aging population, associated with the improving quality of life and consequently an increasing life expectancy, carries with it major challenges regarding the maintenance of the quality of life of elder people. Diseases become more likely, and among these are the ones related to the cardiovascular system, which are rated as some of the leading causes of death worldwide.

A product that allows its user to maintain a normal day-to-day routine while monitoring biomedical parameters, providing constant assessment of their state of health, is a viable answer to need of frequent evaluation of a person's health. This project aims to develop tools to be used in an existent product of this nature named OneCare, owned by Intellicare, a spin-off of the ISA - Intelligent Sensing Anywhere, S.A..

Various portable sensors were used to monitor biomedical signals, in order to obtain vital parameters that indicate whether an individual is in a normal state or suffering from an abnormal situation. The sensors send their data over the Internet to a remote program, which receives the data and stores it. A second program will then allow the graphical visualization of the data received. This project focused on designing, developing and testing the programs responsible for the reception, decoding, storage and graphic representation of the biomedical data.

Keywords: Java, MQTT, JSF, Data Storage, Remote Monitoring of Vital Signs

Índice

Lista de Figuras

Lista de Tabelas

Acrónimos

API – Application Programmer Interface

CSV – Comma Separated Values

ECG – Eletrocardiograma

EDA - Electrodermal Skin Activity

EEG – Eletroencefalografia

EMG - Eletromiografia

IDE - Integrated Development Environment

IMEI - International Mobile Equipment Identity

JDK – Java Development Kit

JEE – Java Enterprise Edition

JPEG - Joint Photographic Experts Group

JRE – Java Runtime Environment

JSF – Java Server Faces

JSON – JavaScript Object Notation

LED – Light-Emitting Diode

M2M – Machine to Machine

MQTT – Message Queuing Telemetry Transport

OP – Oximetria de Pulso

PDF - Portable Document Format

PNG - Portable Network Graphics

sEMG – Eletromiografia de Superfície

SDK – Software Development Kit

SQL – Structured Query Language

SSL - Secure Sockets Layer

SVG - Scalable Vector Graphics

1 Introdução

1.1 Motivação

A saúde é uma das áreas do saber que mais avanços têm sofrido nas últimas décadas. Novos fármacos, novas técnicas e novas tecnologias têm possibilitado um aumento da esperança média de vida [1] bem como da qualidade da mesma. A prevenção assume um papel cada vez mais importante neste aumento da esperança média de vida, acompanhado de qualidade de vida igualmente alta. O facto de ser possível hoje em dia utilizar dispositivos que permitam às pessoas realizar o seu dia-a-dia com normalidade e ao mesmo tempo monitorizar a sua saúde é uma grande contribuição para a manutenção de qualidade de vida elevada, pelo que esta é uma área suscetível a avanços quer no campo da versatilidade, quer do conforto ou da fiabilidade.

De acordo com dados do INE e da PORDATA, correspondentes a informação relativa a Portugal, a esperança média de vida à nascença situa-se nos 79,5 anos, o valor mais alto desde que há registo [1]. Por sua vez, a esperança média de vida para uma pessoa com 65 também se encontra no valor mais alto alguma vez registado, 18,6 anos [2]. A tendência tem sido de aumento consecutivo ano após ano, o que traduz o aumento da qualidade de vida já mencionado.

O número de óbitos em Portugal desde a década de 1990 tem-se situado sempre entre os 101990 (em 2006) e os 108095 (em 2003) [3], o que corresponde a aproximadamente 1% da população portuguesa existente no ano respetivo [4]. Do total, a principal causa de morte em Portugal têm sido doenças do sistema cardiovascular com uma percentagem de 30,7%, embora com tendência a diminuir, sendo que o cancro tem visto a sua percentagem subir (situava-se nos 24,8% em 2011) [5]. Considerando o aumento da esperança média de vida, e tendo em conta a incidência crescente de doenças cardiovasculares com o aumentar da idade, a monitorização de casos de risco surge como uma resposta eficaz de deteção de anomalias em sinais vitais, que pode salvar vidas. É por isso mesmo uma área da saúde em expansão e que merece investimento. O estudo de parâmetros que permitam esta deteção é crucial para o desenvolver de um produto prático e funcional.

A Engenharia Biomédica afirma-se como uma área de saber e aplicação de conhecimentos que visa melhorar a qualidade de vida da Humanidade. Criando novos instrumentos capazes de desempenhar funções até hoje apenas imaginadas, desenvolvendo novas técnicas a ser aplicadas na área da Medicina, indo mais longe na busca da regeneração e substituição de tecidos. Existem uma miríade de possibilidades em que o conhecimento adquirido e utilizado nesta área pode contribuir para uma melhor vida da população.

Na senda desta busca de melhor qualidade de vida, associada a um estilo de vida mais saudável, surgem dispositivos de monitorização de parâmetros biomédicos que permitem ao seu utilizador ter um maior controlo sobre a sua saúde. A evolução tecnológica tem permitido a criação de dispositivos cada vez mais pequenos, portáteis, mais eficazes, mais versáteis e acima de tudo mais confortáveis e fáceis de utilizar. Existem já dispositivos que enviam remotamente as monitorizações para que, em tempo quase real, possam ser visualizadas noutra parte do Mundo, fazendo uso da Internet. Sendo uma área de instrumentação, a área de monitorização remota envolve várias áreas do saber: exige dispositivos evoluídos do ponto de vista eletrónico, programas capazes de enviar, receber e analisar sinais vitais, em tempo útil, e sem consumir grandes recursos, médicos para confirmar que os dados são válidos, entre outros detalhes

necessários para garantir que este tipo de tecnologia é de facto um passo em frente na qualidade de vida e saúde da população.

Já existem diversos produtos, quer comerciais de acesso ao público geral, quer para utilização clínica, que procuram suprir estas necessidades de mercado. Entre estes produtos destacam-se os monitores Holter, o principal método clínico para monitorização do sistema cardiovascular (em utilização desde a década de 1960), e diversos produtos comerciais como o *Corventis AVIVO*, o *Toumaz SensiumVitals*, o *iRythm Zio Patch*, entre outros discutidos mais adiante, no estudo do estado da arte. Estes produtos comerciais medem sinais vitais e analisam os mesmos (alguns em tempo real como o *AVIVO*, outros pós-utilização nos laboratórios da empresa como o *Zio Patch*), detetando anomalias. O sinal fundamental e base de todos os produtos é o ECG (a partir do qual se extraem valores importantes para diagnóstico como a Frequência Cardíaca e a Variabilidade de Frequência Cardíaca), mas outros sinais são também monitorizados para complementar a informação. Alguns exemplos destes sinais também medidos são a Taxa Respiratória, a Temperatura, a Saturação de Oxigénio no sangue, a Atividade Física e Postura e Fluidez do Sangue.

Nesse sentido, a presente tese pretende refletir o desenvolvimento de um produto que possa competir com estes a nível de funcionalidade e qualidade, acrescentando valor à solução OneCare já existente.

1.2 Contexto

O projeto desenvolvido visa a conclusão do Mestrado Integrado em Engenharia Biomédica, no ano letivo de 2012/2013, a cargo da Universidade de Coimbra.

O mesmo está inserido no desenvolver do produto OneCare da Intellicare, uma *spin-off* da ISA – Intelligent Sensing Anywhere, S.A. focada na aplicação do vasto conhecimento tecnológico da ISA na área da saúde [6]. A ISA é uma empresa com larga experiência em telemetria, aplicando sistemas que envolvem transmissão remota de dados em várias áreas como a energia, a saúde, a domótica ou a segurança. Possui também muita experiência em projetos de investigação, conjugando a vertente de desenvolvimento de novos produtos e tecnologias com a parte comercial, estando neste momento estabelecida nacional e internacionalmente.

O OneCare é um dos produtos da Intellicare, inserido no mercado da instrumentação para a saúde, com especial enfâse na monitorização remota de parâmetros biomédicos. Este produto visa coletar dados de vários sinais biomédicos e disponibiliza-los num portal web para posterior acesso em qualquer localização com acesso à Internet. O presente projeto insere-se no desenvolvimento de funcionalidades adicionais para este produto tornando-o mais completo e útil para a qualidade de vida do seu utilizador. Os detalhes desta parceria, da gama de produtos OneCare, bem como uma análise do mesmo e daquilo que de facto pode ser acrescentado estão discriminados na secção 3.1 deste documento.

1.3 Objetivos

O objetivo principal deste projeto é o acrescer de soluções ao produto OneCare já existente. Pretende-se desenvolver um programa que permita receber e armazenar vários tipos de dados biomédicos. Os dados ficam depois disponíveis para acesso posterior por outra ferramenta também a desenvolver. Estas duas ferramentas acrescentam novas capacidades ao OneCare, que o tornam mais evoluído, competitivo, e robusto no que toca às funções que desempenha.

A integração de tecnologias atuais e em constante desenvolvimento, como é o caso da linguagem de programação Java e do protocolo de comunicação MQTT, é um tópico essencial ao desenvolvimento do projeto, uma vez que se pretende o produto final esteja na vanguarda

das opções disponíveis para o ramo da monitorização móvel. Estas tecnologias permitirão a comunicação entre máquinas com acesso à Internet, e assim acrescentarão novas funcionalidades ao produto final.

1.4 Organização da Tese

A tese divide-se em 7 capítulos, com diferentes objetivos, descritos abaixo.

Neste capítulo (Capítulo 1), é feita a introdução ao tema, descritas as motivações por detrás do projeto e os objetivos do mesmo.

O Capítulo 2 descreve a fundamentação teórica, tanto da fisiologia do organismo humano como do funcionamento dos materiais e tecnologia utilizada.

O Capítulo 3 apresenta o Estado da Arte de toda a tecnologia associada a este projeto, descrevendo os produtos e opções mais atuais, as suas funcionalidades e assim proporcionando a compreensão por detrás da sua utilização.

O Capítulo 4 visa descrever a Arquitetura do Projeto, facilitando deste modo a compreensão da metodologia utilizada, de considerações que foram tidas em conta aquando do desenvolvimento do projeto, e disponibilizando uma visão global do mesmo.

O Capítulo 5 descreve detalhadamente a Metodologia utilizada, apresentando o trabalho desenvolvido e os procedimentos por detrás dos resultados finais.

O Capítulo 6 apresenta os Resultados obtidos e a Discussão inerente aos mesmos e informações obtidas aquando do desenvolvimento do projeto.

No Capítulo 7 enumeram-se as conclusões obtidas ao longo projeto, avaliam-se os resultados e lançam-se as bases para trabalho futuro.

Após os Capítulos principais, existem ainda 4 Anexos com informação complementar para melhor compreensão deste projeto.

2 Fundamentação Teórica

Neste capítulo serão apresentados os conceitos fisiológicos e tecnológicos cujo conhecimento é necessário para uma correta compreensão do projeto.

2.1 Fisiologia

Antes de iniciar a descrição do projeto, é importante compreender o contexto biomédico em que este projeto se insere. De seguida são assim apresentados o Sistema Cardiovascular Humano e o Sistema Respiratório, que devido à sua grande importância no organismo Humano terão na sua monitorização o foco biomédico deste projeto. Além destes Sistemas, serão apresentados os exames médicos que permitem a análise do estado de saúde dos mesmos, e que serão a base dos dados biomédicos que o projeto visa coletar, tratar e disponibilizar para análise.

2.1.1 Sistema Cardiovascular Humano

O organismo humano possui um sistema circulatório capaz de distribuir, recolher e transportar substâncias no seu interior. Este sistema serve-se de vasos preenchidos por um fluido para desempenhar as suas funções, podendo ser dividido consoante o fluido circulante, sendo que os dois sistemas têm propósitos diferentes [7]:

- Sistema Cardiovascular (em que o fluido é o Sangue)
- Sistema Linfático (em que o fluido é a Linfa).

O Sistema Cardiovascular é constituído pelos vasos que transportam o sangue e pelo coração. O coração é um órgão do organismo humano que proporciona a circulação contínua de sangue através do mesmo organismo [8]. O seu funcionamento é de bomba aspirante-premente, com 4 cavidades, separadas duas a duas.

É frequente falar num coração esquerdo e coração direito, dado que os dois lados do coração não comunicam. Os dois lados separados fazem parte de duas circulações: o lado direito da circulação pulmonar (ou pequena) e o lado esquerdo da circulação sistémica (ou grande).

2.1.1.1 Circulação

A função da circulação é servir as necessidades dos diferentes tecidos do organismo, garantindo um ambiente propício para a sobrevivência e funcionamento ótimo das células. Para tal, permite transportar nutrientes (e outras substâncias) para os tecidos, bem como produtos metabólicos dos tecidos e hormonas [9]. Podemos assim considerar que a função principal da circulação é de transporte e manutenção da homeostasia, pelo que o fluxo sanguíneo é adaptado às necessidades do organismo.

2.1.1.2 Circulação Sistémica (ou Grande Circulação)

A Circulação Sistémica percorre todo o organismo, de modo a desempenhar as suas funções em todos os tecidos do mesmo. É nesta circulação que o sangue percorre todos os tecidos do organismo, libertando Oxigénio (O_2) para as células, efetuando distribuição de nutrientes, e coletando resíduos da sua atividade.

2.1.1.3 Circulação Pulmonar (ou Pequena Circulação)

A Circulação Pulmonar tem como objetivo levar o sangue até ao interior dos pulmões, para que aqui o sangue liberte Dióxido de Carbono e receba Oxigénio, num processo denominado trocas gasosas. Esta circulação inicia-se no ventrículo direito, em que o sangue venoso, rico em CO_2 sai pela artéria pulmonar em direção aos Pulmões. Após uma série de divisões, o sangue chega aos alvéolos pulmonares em capilares extremamente finos, para que seja possível a difusão de gases através das paredes dos capilares. A rede de capilares drena depois para as veias

pulmonares, que por sua vez ligam à aurícula esquerda. O sangue arterial, rico em Oxigénio, é assim encaminhado para o início da circulação sistémica.

2.1.1.4 Diagrama elétrico do Coração

A descarga elétrica em cada ciclo cardíaco começa normalmente numa zona da aurícula direita chamada Nodo Sinoatrial. A partir desta zona a descarga propaga-se através das fibras musculares da aurícula. Posteriormente há um atraso na propagação ao longo de uma zona denominada Nodo Atrioventricular, a partir do qual a descarga se propaga rapidamente através de tecido especializado, primeiro através do Feixe de His, que posteriormente se divide ao longo do septo em dois feixes, direito e esquerdo. No final dos feixes, na zona ventricular, a condução dá-se ao longo de Fibras de Purkinje, especializadas para o efeito. De notar que o início da propagação pode não ocorrer no Nodo Sinoatrial [10].

2.1.2 Sistema Respiratório

O Sistema Respiratório é responsável pela renovação do ar nos pulmões, permitindo assim ao sangue efetuar trocas gasosas, principalmente libertando Dióxido de Carbono e captando Oxigénio. É constituído pelo trato respiratório superior (cavidade nasal, boca, laringe e faringe), responsável pelo aquecimento, humidificação e filtração do ar, e trato respiratório inferior (traqueia, brônquios principais e ramificações, Pulmões).

A importância deste sistema prende-se com o seguinte ciclo do organismo humano:

1. Ventilação (mobilização do ar para dentro e fora dos Pulmões);
2. Trocas gasosas de O_2 e CO_2 através da membrana respiratória, por difusão;
3. Transporte sanguíneo de O_2 e CO_2 para todas as células do organismo;
4. Trocas gasosas de O_2 e CO_2 nos tecidos;
5. Utilização celular de O_2 e produção celular de CO_2.

Ou seja, através do aparelho respiratório são expelidos produtos da atividade celular, ao mesmo tempo que se fornece ao organismo um dos *combustíveis* fundamentais ao bom funcionamento celular, o Oxigénio.

A unidade funcional dos Pulmões são os Alvéolos Pulmonares, que constituem pequenos *sacos* nas extremidades das ramificações finais das vias respiratórias. Os Alvéolos são estruturas extremamente finas e vascularizadas, que permitem a troca de gases entre o sangue e o ar que os preenche. É nesta zona do Sistema Respiratório que ocorrem a maioria das trocas gasosas.

2.1.2.1 Ventilação

A ventilação pulmonar é o processo através do qual o ar entra e sai dos Pulmões. É assim fundamental ao bom funcionamento do Sistema Respiratório, e vê a sua atividade dividida em duas etapas, inspiração e expiração:

- **Inspiração**: processo através do qual é promovida a entrada de ar do exterior do organismo para as vias respiratórias, e consequentemente para os Pulmões. Para tal, a caixa torácica expande, diminuindo a pressão interna e tornando-a inferior à pressão do ar no exterior do corpo. Assim, o ar irá entrar nas vias respiratórias.
- **Expiração**: processo através do qual o ar é expelido dos Pulmões. O inverso da Inspiração, na Expiração a caixa torácica comprime, aumentando a pressão alveolar e assim forçando o ar a sair dos pulmões.

De referir que para atingir as pressões necessárias à ventilação, existem vários músculos associados ao Sistema Respiratório cuja atividade é responsável pelas variações de pressão.

A taxa de ventilação é proporcional à necessidade de renovação de ar nos Pulmões, que por sua vez depende do estado do organismo e das necessidades do mesmo.

2.1.2.2 Regulação da respiração

A Respiração é regulada pelo centro Respiratório, um grupo de neurónios localizados no Tronco Cerebral (mais concretamente, no Bolbo Raquidiano) e que estimula os músculos respiratórios. Esta é a principal área responsável pelo controlo da respiração. No entanto existem outras áreas que podem influenciar a Respiração, como a Ponte, também no Tronco Cerebral.

Um detalhe muito importante (e explorado neste projeto) é o de que a Taxa Respiratória é um indicador importante acerca do organismo. A sua regulação é influenciada por vários fatores e reflete diferentes estados do organismo, para além de possíveis doenças ou anormalidades do Sistema Respiratório. Por exemplo um esforço físico é acompanhado de um aumento da Taxa Respiratória, devido à maior necessidade de Oxigénio para utilização pelas células. Conjugando o conhecimento da Taxa Respiratória com outros dados (como um Eletrocardiograma), é possível retirar conclusões mais válidas e concretas sobre um organismo em estudo.

2.1.3 ECG – Eletrocardiograma

O Eletrocardiograma (ECG) é um dos exames mais utilizados no dia-a-dia clínico, seja por precaução, monitorização de saúde, durante cirurgias ou no período pós-cirurgia. O facto de permitir extrair variada informação acerca do estado do Coração, órgão vital, e ser não invasivo, rápido e indolor, torna este um dos principais e mais comuns exames a ser realizados.

O ECG mede a atividade elétrica do coração, através de vários elétrodos colocados à superfície da pele do paciente. Diferentes padrões podem ser obtidos, recorrendo a diferentes localizações dos elétrodos, mas os resultados finais devem ser os mesmos.

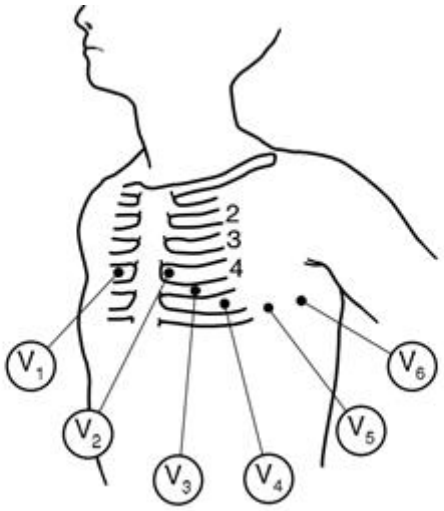


Figura 1: ECG - Posicionamento dos elétrodos no peito [10]

Os elétrodos comunicam depois com um dispositivo central que coleta os dados e faz a disponibilização dos resultados.

Através dos resultados obtidos, pode ser analisada a forma do diagrama elétrico do Coração, que se comparado com casos conhecidos pode levar ao diagnóstico de complicações de saúde. Ao mesmo tempo, permite o cálculo da frequência cardíaca e da sua variabilidade, que também são duas ferramentas importantes no diagnóstico de situações anómalas à perfeita saúde de um indivíduo.

Figura 2: Exemplo de ECG [10]

Cada vez mais surgem aparelhos que permitem efetuar Eletrocardiogramas confortavelmente e em diversas situações, como na rotina diária, na prática de desporto, a dormir, etc. Aparelhos deste tipo serão discutidos no Estado da Arte (secção 3.3) e constituem uma importante ferramenta no que toca à monitorização remota do estado de saúde.

Por ser uma ferramenta poderosa e versátil, o Eletrocardiograma é um dos exames a incluir no projeto que visa a monitorização do estado de saúde de um indivíduo.

2.1.3.1 Forma do ECG

Devido à menor massa muscular das aurículas quando comparada com a dos ventrículos, a alteração de carga elétrica que acompanha a contração das aurículas é menos intensa, o que se traduz num sinal mais fraco. À contração auricular corresponde a onda P no ECG (ver Figura 3).

Posteriormente, como a massa muscular dos ventrículos é manifestamente maior, a despolarização ventricular leva a uma alteração significativa do sinal do ECG, denominada por complexo QRS (ver Figura 3). Finalmente, o regresso da massa ventricular ao seu estado elétrico inicial (ou seja, a repolarização ventricular) leva ao aparecimento de um sinal (de menor intensidade que o complexo QRS) que no ECG se denomina por onda T (ver Figura 3). O intervalo entre as ondas S e T é chamado segmento ST.

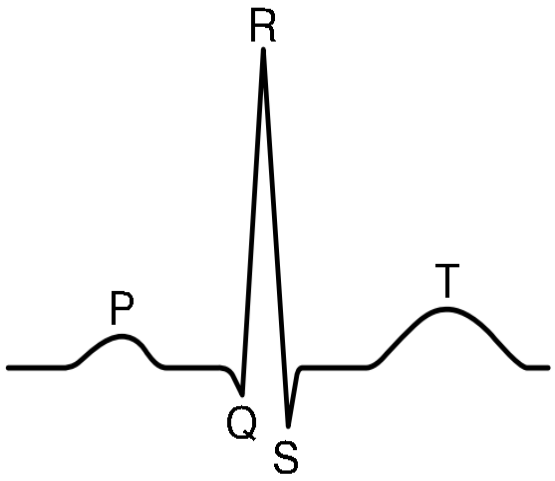


Figura 3: Sinal Típico de ECG [10]

As diferentes partes do complexo QRS são nomeadas do seguinte modo:

- Se a primeira deflexão for para baixo, esta é chamada de onda Q;
- Onda R é uma deflexão para cima, quer seja precedida por uma deflexão para baixo, ou não;
- Onda S é qualquer deflexão para baixo, após a onda R, tenha ou não a onda R sido precedida por uma onda Q.

As letras que definem cada deflexão (chamadas ondas) foram escolhidas aleatoriamente aquando do início do estudo desta técnica, não tendo qualquer significado físico ou fisiológico.

2.1.3.2 Tempos e Velocidades

O sinal captado pelos dispositivos de ECG é tradicionalmente registado numa folha de papel que se desloca sob uma caneta. Essa caneta move-se e traça na folha em movimento a linha correspondente ao sinal obtido [10]. Atualmente já existem ECG's digitais, que guardam e processam o sinal obtido digitalmente, sendo posteriormente visualizado o resultado num *display* próprio ou num dispositivo que tenha acesso aos dados (monitor de um computador, *tablet*, etc.) [11].

Nos sistemas de visualização tradicional, o resultado do exame assenta numa base dividida em quadrados grandes e pequenos. Na horizontal, os grandes representam intervalos de 0,2 segundos (200 milissegundos - o que implica que 1 minuto são 300 quadrados grandes), enquanto os pequenos representam intervalos de 0,04 segundos (40 milissegundos) [10]. Assim é possível inferir sobre as taxas a que os diferentes eventos cardíacos ocorrem no organismo.

A frequência cardíaca é o número de vezes que um ciclo cardíaco completo ocorre durante um minuto. Medindo a distância temporal entre dois picos eventos semelhantes (por exemplo picos R-R, que são os mais destacáveis no sinal de ECG) é possível definir a frequência cardíaca. Dividindo 300 pelo número de quadrados grandes que separam dois picos R-R irá resultar na frequência cardíaca do indivíduo em causa.

Simultaneamente, outros tempos podem ser calculados. O intervalo PR (medido desde o início da onda P até ao início do complexo QRS) representa o tempo necessário para que a despolarização se propague desde o Nodo Sinoatrial até ao músculo ventricular. Os valores típicos para este intervalo são de entre 0,12s a 0,2s, ou seja, três a cinco quadrados pequenos. A maior parte deste intervalo corresponde ao atraso que ocorre no Nodo Atrioventricular. Um valor demasiado baixo deste intervalo pode significar que o início da despolarização ocorreu muito perto do Nodo Atrioventricular ou que há uma condução anormalmente rápida do sinal elétrico da aurícula até ao ventrículo.

A duração do complexo QRS ilustra o tempo que a excitação demora a espalhar-se pelos ventrículos, sendo normalmente da ordem dos 0,12s (três quadrados pequenos) ou menor. Anomalias na condução traduzem-se em atrasos da mesma, o que leva a um complexo QRS mais largo [10].

2.1.4 Oximetria de Pulso

O Oxigénio é um elemento fundamental ao funcionamento de cada célula do organismo humano, sendo que na ausência prolongada de Oxigénio as células acabam por morrer. Assim, a distribuição celular de Oxigénio é um indicador muito importante do estado de saúde de um indivíduo. Um dos métodos mais utilizados para inferir sobre este fator é a Oximetria de Pulso, um exame de caracter não invasivo [12].

A distribuição de Oxigénio pelas células serve-se principalmente de dois sistemas do organismo humano: o Sistema Respiratório e o Sistema Cardiovascular. Tudo se inicia com a ventilação pulmonar, em que o ar entra e sai dos pulmões onde decorrem as trocas gasosas. Nestas trocas, Oxigénio do ar difunde para o sangue e no sentido inverso segue o Dióxido de Carbono (descrito na secção 2.1.1.3). O sangue oxigenado volta ao coração para depois percorrer a Circulação Sistémica distribuindo o Oxigénio pelas células e recolhendo produtos da sua atividade.

A Oximetria de Pulso baseia-se em duas técnicas: a Espectrofotometria, para obter os valores de Saturação de Oxigénio no sangue arterial, e a Fotoplestismografia, para calcular a Frequência Cardíaca (e possivelmente a Pressão Arterial).

A Espectrofotometria consiste em utilizar uma fonte de luz de energia conhecida, fazê-la passar pelo meio cuja composição se pretende medir, e captar a luz transmitida por esse meio. Analisando o espectro da luz captada, e comparando com o espectro original, pode obter-se a composição qualitativa e quantitativa desse meio. É exatamente esse o procedimento que se efetua neste caso: utiliza-se tradicionalmente a ponta do dedo, onde se coloca um dedal adaptado, que de um dos lados tem fontes de luz (normalmente duas – uma vermelha de comprimento de onda 660 nanómetros e outra infravermelha de comprimento de onda cerca dos 910/940 nanómetros – para as quais são utilizados *LED's* – Light-Emitting Diode ou Díodo Emissor de Luz em português) e do outro um fotodetetor que capta a luz proveniente das fontes e que passa através do dedo, e consequentemente dos vasos sanguíneos.

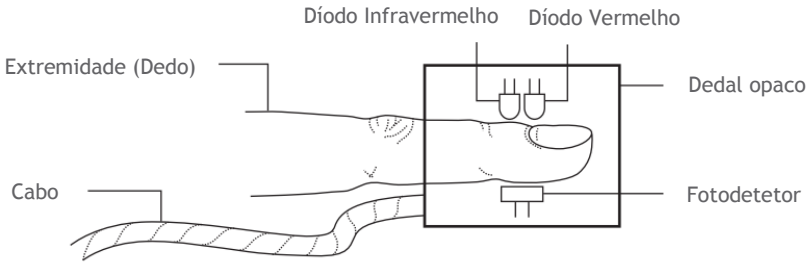


Figura 4: Componentes de um Oxímetro de Pulso [13]

Posteriormente, um conjunto de circuitos eletrónicos garantem a aquisição de um bom sinal e a sua análise. Comparando a luz captada com a originalmente emitida (e cujas propriedades são conhecidas), é possível calcular a Saturação de Oxigénio no Sangue Arterial, isto porque são já conhecidos os espectros de absorção da Hemoglobina Oxigenada (HbO_2) e Desoxigenada (Hb).

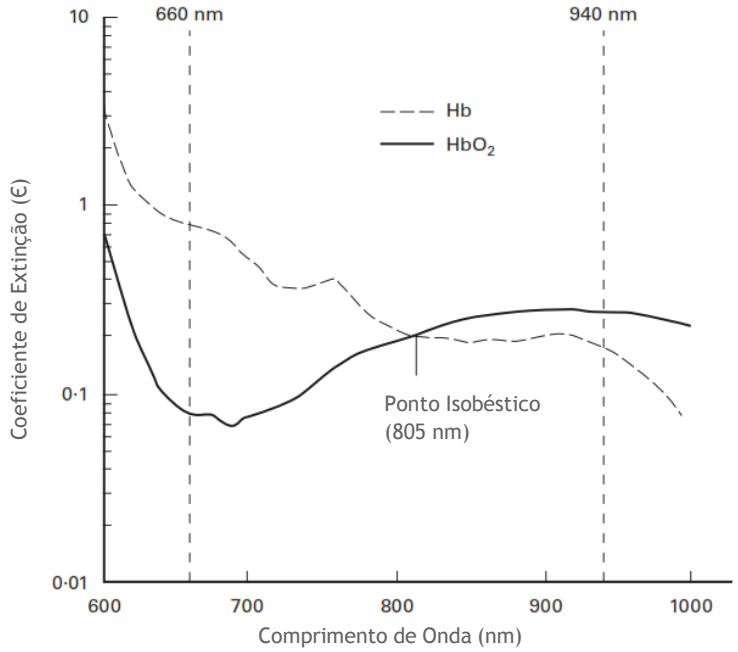


Figura 5: Espetro de absorção da hemoglobina oxigenada e desoxigenada [13]

Através de um Oxímetro de Pulso é assim possível conhecer a Saturação de Oxigénio no Sangue Arterial. Oxímetros de Pulso são muito utilizados em meio hospitalar, por exemplo durante e pós cirurgias, uma vez que são um indicador importante do estado de saúde de um indivíduo. Os valores de Saturação de Oxigénio no Sangue Arterial devem estar acima de 95%. Caso contrário, e se não há nenhum defeito no exame ou no aparelho, algo de anormal se deverá estar a passar com o utilizador do dispositivo [13].

A Fotoplestimografia permite a um Oxímetro de Pulso calcular a Frequência Cardíaca e também a Pressão Arterial (embora não seja muito utilizada esta última funcionalidade). A Fotopletismografia consiste em calcular o volume de um espaço através do estudo da atenuação da luz que passa através do mesmo. Fazendo uso desta técnica e dos restantes recursos que possui, um Oxímetro de Pulso pode assim extrair mais informação médica complementar.

A variação de volume sanguíneo que ocorre aquando da sístole e diástole cardíacas faz com que a atenuação da luz que passa através dos vasos se altere. Essa variação é utilizada pelo Oxímetro de Pulso para calcular a Frequência Cardíaca e também para validar o seu funcionamento, ou seja, para se auto testar e garantir que está a funcionar corretamente.

2.2 Tecnologias de Informação e Comunicação

Sendo este um projeto de natureza tecnológica, uma apresentação dos recursos utilizados no decorrer do trabalho revela-se pertinente para facilitar a sua compreensão. Nesse sentido, serão apresentados os fundamentos tecnológicos cujo conhecimento é necessário para que o desenvolvimento do projeto seja devidamente compreendido, bem como todas as escolhas tecnológicas no seu decorrer. Entre os recursos tecnológicos utilizados encontra-se a linguagem de programação Java e o protocolo de comunicação *Message Queuing Telemetry Transport* (MQTT) que serão explicados mais adiante.

2.2.1 Java

Java é uma linguagem de programação orientada a objetos surgida em 1995, que assumiu uma evolução constante e se estabeleceu como a principal linguagem utilizada em desenvolvimento Web. No entanto a sua aplicação abrange mais áreas, entre elas aplicações móveis (por exemplo em *Android*), jogos, *software* empresarial, entre outras [14].

Algumas considerações devem ser tidas em conta relativamente a Java para que se possa compreender completamente este projeto, sem ser necessário ter conhecimentos avançados desta linguagem.

Java é uma linguagem orientada a objetos[1], o que significa que Java é baseada na criação e utilização destas entidades. Cada objeto é uma instância de uma classe (i.e. criado a partir de uma classe), o que torna Java numa linguagem baseada em classes.

Classes são a essência de Java, constituindo as fundações sobre as quais toda a linguagem é construída, pois definem a natureza dos objetos. São por isso a base da programação orientada a objetos em Java. Numa classe são definidos dados e código, que atua sobre esses dados. Este código é contido em métodos, que funcionam como ações das classes.

As classes funcionam como um modelo que define a forma de objetos, especificando os dados e métodos que irá possuir esse mesmo objeto. São utilizadas para criar os objetos e até serem criados os objetos, as classes não passam de uma abstração lógica. Apenas quando um objeto baseado numa dada classe é instanciado, é que se forma uma representação física da classe

[1] Objetos são entidades que possuem uma identidade, atributos (campos de dados que definem o objeto) e métodos que efetuam procedimentos.

escrita. Os dados e métodos que constituem uma classe são denominados membros da classe, ou variáveis da instância. Várias classes podem ser agrupadas em *packages* para melhor funcionamento e estruturação do programa no qual se inserem.

Java *packages* não são mais do que conjuntos de classes. Uma Java *package* é assim tipicamente utilizada para organizar as classes pertencentes à mesma categoria, ou que desempenham funcionalidade semelhante. As *packages* facilitam a organização das classes, sendo possível a sua agregação consoante o contexto em que foram escritas, ou seja, consoante o objetivo funcional das mesmas.

Outro pormenor importante é a existência de *packages* já completas e funcionais, disponíveis para *download* e acesso por qualquer programador. Este tipo especial de *package* é denominado livraria e possui classes pré-definidas que executam determinadas tarefas. As livrarias facilitam a escrita de um programa, bastando que o programa importe uma livraria para que este possa aceder imediatamente a todas as funcionalidades que a mesma disponibiliza.

Em suma, um programa está frequentemente dividido em *packages* que facilitam a sua organização funcional. Estas possuem classes, que por sua vez são o esqueleto do programa, contendo todas as funções desejadas bem como variáveis para armazenamento interno de dados. Um objeto apenas é criado quando uma classe é instanciada, sendo que cada classe pode ser instanciada várias vezes. O objeto terá todas as variáveis e métodos da classe que o originou, sendo assim uma representação física do código definido na classe.

Para programar em Java são frequentemente utilizados IDE's - Ambiente de Desenvolvimento Integrado (em inglês, *Integrated Development Environment*), que simplificam a programação, providenciando uma interface gráfica mais intuitiva a este efeito [15].

2.2.2 *JSON* – JavaScript Object Notation

JSON é um formato simples de intercâmbio de dados computacionais. Apesar de baseado na notação de JavaScript, não requer necessariamente o uso de JavaScript para a sua utilização, sendo por exemplo possível de utilizar entre programas escritos em Java [16].

No desenvolvimento deste projeto foi utilizada esta tecnologia sob a forma *String JSON*. Neste formato, o objeto *JSON* consiste numa *String* estruturada seguindo o esquema da Figura 6:

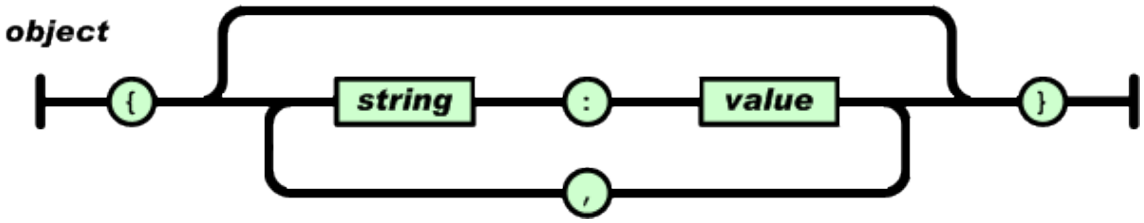


Figura 6: Estrutura de String JSON [16]

Clarificando o conteúdo da Figura 6, o objeto consiste numa *String* (um campo de texto) cuja informação está contida entre duas chavetas. No interior das chavetas podem existir vários campos de dados, separados por vírgulas. O título identificativo de cada campo é uma *String*, seguida de dois pontos (:) e do valor associado a esse campo.

Por exemplo, para guardar a informação "Temperatura" com um valor inteiro de 37, e "Nome" como sendo "Carlos", a *String JSON* seria {"Temperatura":37, "Nome":"Carlos"}.

A sua simplicidade e pequeno tamanho tornou este tipo de objetos populares entre programas informáticos. Por estas razões, e por ser eficaz a transmitir informação sem complexidade

acrescida, esta será a codificação utilizada no projeto global para enviar dados obtidos a partir de sensores biomédicos, e receber os mesmos noutra entidade computacional.

Para criar este tipo de objeto existem já livrarias disponíveis, de acesso sem limitações e que possuem várias funcionalidades uteis, facilitando assim a elaboração de programas mais organizados e funcionalmente estáveis.

2.2.3 MQTT - *Message Queuing Telemetry Transport*

MQTT é um leve[2] protocolo de comunicação Máquina a Máquina (*Machine to Machine ou M2M*) utilizado para publicar ou subscrever a receção de mensagens, entre diferentes dispositivos. Inventado por *Andy Stanford-Clark* da *IBM*, e *Arlen Nipper* na altura (em 1999) da *Arcom* (adquirida em 2006 pela *Eurotech*) e originalmente chamado *MQ Integrator SCADA Device Protocol*, o MQTT é desenhado para ser aberto (i.e. *open source*), simples e fácil de implementar, permitindo milhares de clientes suportados num único servidor. Estas características tornam-no ideal para aplicação em ambientes constrangidos ou de redes com baixa largura de banda, recursos de processamento limitados, pequena memória e valores de latência altos. A arquitetura do MQTT minimiza os requisitos de largura de banda da rede, ao mesmo tempo que garante fiabilidade na entrega e receção de dados [17]. Este protocolo providencia assim um método leve[2] de transporte de mensagens, o que o torna adequado ao conceito da *M2M* e à sua aplicação em comunicação entre sensores de baixa energia, dispositivos móveis, computadores embebidos ou microcontroladores [18].

O funcionamento do MQTT baseia-se no conceito de publicação/subscrição de mensagens em tópicos, recorrendo a um intermediário denominado *broker*. Um *broker* é *software* que funciona como entidade intermédia, capaz de receber mensagens e de as encaminhar para o seu destino. Assim, este *software* faz a gestão das mensagens envolvidas numa comunicação, bem como dos intervenientes da mesma – as entidades que que enviam e recebem as mensagens. Os intervenientes na comunicação são denominados clientes. Um cliente MQTT que subscreve a um tópico num *broker*, recebe todas as mensagens publicadas nesse mesmo tópico. Do mesmo modo, uma mensagem publicada num dado tópico é recebida por todos os clientes que subscreveram a esse tópico. Assim, o funcionamento deste protocolo tem 2 intervenientes fundamentais, sendo que um destes se subdivide em 2:

- **Servidor (*broker*)**: o *broker* é o intermediário da comunicação. É neste que os tópicos são listados e armazenados, e as subscrições ou publicações em cada tópico são endereçadas e registadas. Deve assim estar alojado num servidor, para tornar possível o acesso remoto por parte dos Clientes (sejam eles subscritores ou publicadores).
- **Cliente**:
    - Subscritor: subscreve tópicos, recebendo assim todas as mensagens que são publicadas nesse tópico, enquanto a subscrição for válida.
    - Publicador: publica mensagens nos tópicos-alvo (pode ser um ou vários tópicos).

Os tópicos de MQTT são criados automaticamente quando algum cliente se conecta, não necessitando de qualquer configuração. Por outras palavras, se o cliente *X* subscrever ao tópico *testes*, o tópico *testes* passa a existir sem serem necessários passos adicionais. Além disso, os tópicos podem ser hierarquizados, ou seja, podem ser subdivididos para respeitar uma dada organização. O exemplo que está na documentação do *Mosquitto* [19], um dos *brokers* à

[2] Neste conceito, leve significa que o protocolo consome poucos recursos quer do dispositivo que está a utilizar o MQTT, quer da ligação utilizada para estabelecer a comunicação. Por outras palavras, o MQTT não necessita de grande capacidade de processamento, gasta pouca energia e ocupa uma pequena largura de banda da ligação utilizada.

disposição de utilização, é bem ilustrativo de como se podem organizar os tópicos e será utilizado de seguida para clarificar este assunto.

Supondo que se deseja que vários computadores partilhem a temperatura de um dos seus discos, tal poderá ser feito. O computador cujas temperaturas se desejam medidas necessita apenas de ter a capacidade de saber a temperatura de cada disco, e enviar essa informação através dum cliente MQTT configurado no próprio computador, que publica uma mensagem com a informação de temperatura (ou seja, publica a informação) no seguinte tópico:

- sensors/COMPUTER_NAME/temperature/HARDDRIVE_NAME

Cada computador, ao publicar os dados para o *broker*, substitui no nome do tópico COMPUTER_NAME pelo nome respetivo e HARDDRIVE_NAME pelo nome do disco em análise. Por outras palavras, a informação relativa à temperatura é publicada no subtópico correspondente ao Disco em que a temperatura foi medida, que por sua vez é um subtópico de *temperature*, este sendo um subtópico o nome do computador em causa, que finalmente é um subtópico de *Sensors*.

No exemplo da Figura 7, o computador de nome Casa PC, publica a informação relativa ao seu Disco Principal no tópico DiscoPrincipal, subtópico de *temperature*, que é subtópico de Casa PC, por sua vez subtópico de *Sensors*.

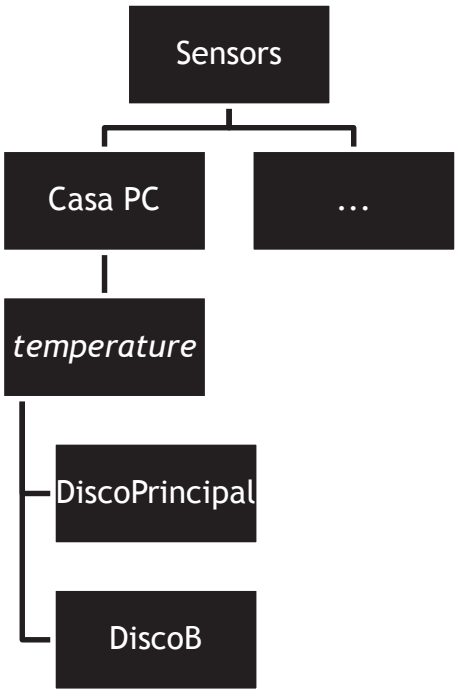


Figura 7: Exemplo de organização de tópicos MQTT

Um cliente configurado noutra máquina pode depois receber as mensagens publicadas pelos diferentes computadores, ao criar subscrições nesses mesmos tópicos. Uma subscrição pode ser apenas dirigida a um tópico específico (neste exemplo, caso se deseje subscrever a apenas um disco rígido de um computador) ou dirigida a vários tópicos (por exemplo se se desejar subscrever a vários discos rígidos do mesmo computador, ou a vários computadores).

A maneira como tal é atingido depende de *broker* para *broker*. Por exemplo no *Mosquitto*, para subscrever a todos os computadores e todos os discos rígidos que publicam o tópico a subscrever seria:

- sensors/+/temperature/# ou
- sensors/+/temperature/+

Para receber todos os discos do Computador *Casa PC* o tópico seria:

- sensors/Casa PC/temperature/# ou
- sensors/Casa PC/temperature/+

Um cliente que deseje subscrever por exemplo ao disco *DiscoPrincipal* do computador *Casa* sobrescreveria ao tópico:

- sensors/Casa/temperature/DiscoPrincipal

A diferença entre o + e o # é que o # apenas subscreve a todos os tópicos do último ramo da hierarquia, enquanto o + pode ser utilizado para subscrever a todos os tópicos da hierarquia ou apenas aos finais.

Um detalhe importante do MQTT é facto de alguns *brokers* permitirem que após a conexão de um cliente a um dado tópico, este receba todos os dados que tenham sido enviados para o mesmo, ainda que tenham sido enviados sem o cliente estar conectado. Se mensagens forem enviadas quando o cliente não está conectado mas este já tiver estado subscrito ao tópico em questão, aquando da re-conexão do cliente ao *broker*, o mesmo vai receber todos as mensagens enviadas entre a última vez que esteve ligado e o momento da nova conexão. Para tal é necessário que o *broker* suporte esta funcionalidade e que o cliente esteja configurado para este efeito.

2.2.4 Big Data

Big Data é uma das mais recentes tendências tecnológicas no que toca ao armazenamento de dados, tendo em conta o aumento de volume de dados que são utilizados diariamente, à velocidade a que os dados são requisitados e à necessidade de encontrar novas soluções para os desafios tecnológicos que isto representa. A grande quantidade e complexidade de dados tornam obsoletos os convencionais métodos de Bases de Dados utilizados até hoje, e respondendo a este problema decorrente da evolução surgem novos métodos de análise, armazenamento e organização de dados. A este conjunto de novas tecnologias chama-se *Big Data*.

Antes de compreender o que é o *Big Data*, convém compreender o que é uma Base de Dados tradicional, e porque é necessária uma nova aproximação a esta questão. Assim sendo, as características essenciais de uma Base de Dados tradicional devem seguir três princípios:

- A Base de Dados deve persistir ao longo do tempo;
- Deve estar estruturada de forma a incluir ferramentas bem definidas e úteis para aceder aos seus dados e extrair qualquer informação nela contida;
- A sua estrutura deve ser muito bem definida (o melhor possível), para permitir perguntas esclarecidas e respetivas respostas concretas.

As Bases de Dados Relacionais baseiam-se no Modelo Relacional. Este prevê uma coleção de tabelas que se relacionam entre si. Uma tabela de dados tem colunas que correspondem a atributos dos dados, e linhas que correspondem a cada entrada de dados. Uma tabela deve possuir uma Chave Primária, que identifica univocamente cada membro desta tabela. Pode também possuir colunas cujos dados são obtidos a partir de outras tabelas, estabelecendo Chaves Estrangeiras (que correspondem a Chaves Primárias de outras tabelas).

Existem vários *softwares* que permite a criação e edição deste tipo de Bases de Dados. Na Figura 8 está um exemplo de Base de Dados relacional para melhor compreensão do conceito.

O exemplo (criado através do programa Microsoft Access) corresponde a uma base de dados de uma loja que vende por encomendas. Cada *Order* (Encomenda) pode ter vários *Order Detail*'s (Detalhes da Encomenda), mas cada *Order Detail* apenas pode corresponder a uma *Order*. *OrderID* é Chave Primária da tabela *Orders* (ou seja, identifica inequivocamente cada *Order*) e Chave Estrangeira na tabela *Order Details*, permitindo estabelecer a comunicação entre as duas tabelas. Por sua vez, cada *Customer* (Cliente) pode fazer várias *Orders*, mas cada *Order* apenas pode ser feita por um *Customer*. Neste caso, a Chave Primária da tabela *Customers* é *CustomerID*, que é também Chave Estrangeira na tabela *Orders*, sendo assim responsável pela ligação entre as duas tabelas.

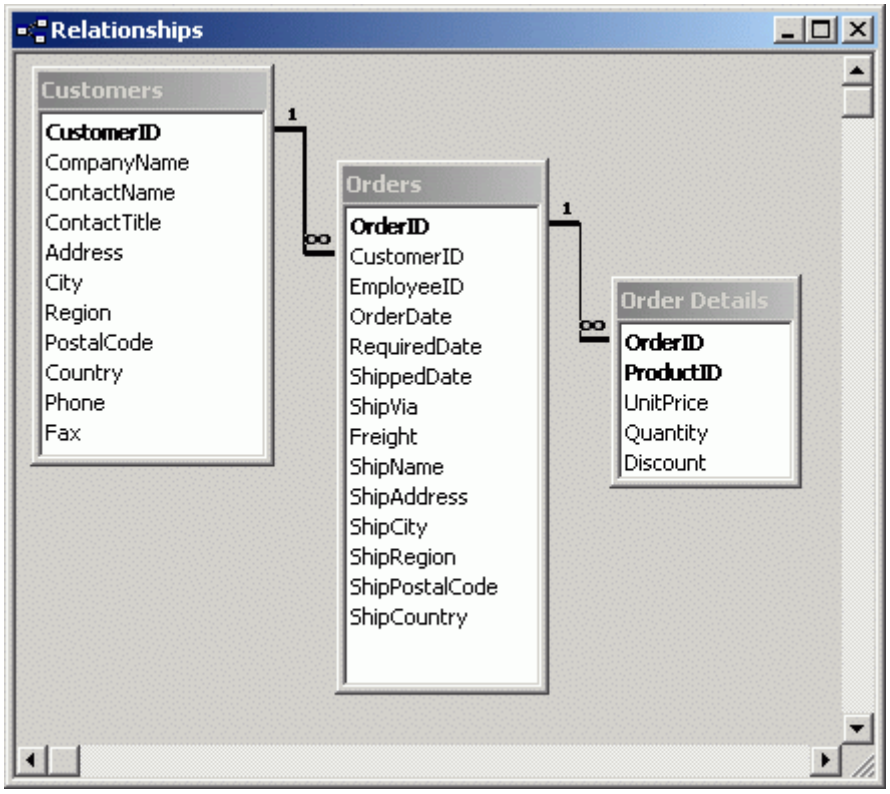


Figura 8: Exemplo de Base de Dados Relacional [20]

Para aceder a dados de uma Base de Dados Relacional é necessário utilizar um Sistema de Gestão de Bases de Dados (*DBMS - Data Base Management System*), que constitui o conjunto de ferramentas que permite armazenar, modificar, e extrair informação da mesma. Diferentes ferramentas possuem diferentes linguagens, contudo existe uma linguagem muito popular chamada SQL – *Structured Query Language*, que permite executar um grande conjunto de procedimentos sobre as Bases de Dados Relacionais.

Atualmente verifica-se que as características dos dados que são criados e postos em circulação no meio tecnológico começam a tornar-se demasiado complexas para estes modelos relacionais poderem extrair o máximo da informação, com um desempenho aceitável. No entanto, não se pense que os conceitos inerentes ao *Big Data* são uma completa quebra em relação aos princípios tradicionais. São antes a evolução natural dos mesmos, para responder a novas exigências.

Big Data não depende apenas do tamanho dos dados, aliás, este até é capaz de ser o fator menos característico dos dados envolvidos, sendo os dois outros fatores a velocidade e variedade. Passando a explicar os 3 V's do *Big Data*:

- **Volume**: hoje em dia são cada vez mais as tecnologias que movem vários Peta-Bytes (Peta = 10^{15}) de dados;
- **Velocidade**: relativa à taxa de variação dos dados, e o quão depressa é necessário processar estes dados para obter valores de interesse real. As tecnologias tradicionais não foram criadas tende em conta este requisito, sendo assim inadequadas ao armazenamento e utilização de dados de alta velocidade. Assim, dados que sejam criados e agregados a uma velocidade elevada, e cuja análise necessita de velocidade alta para que possam ser extraídas conclusões dos mesmos, remetem para um tipo diferente de tecnologia, que não a tradicional;
- **Variedade**: refere-se a vários tipos de dados e ficheiros que se tornam importantes de gerir e analisar cuidadosamente, tarefa para a qual as bases de dados relacionais tradicionais não foram desenhadas, apesar de o poderem cumprir. Exemplos desta variedade são ficheiros de som, vídeo, imagens, documentos, dados de geo-localização, *web logs* e *Strings* de texto [21].

Ao conjunto de ferramentas e tecnologias que se dedicam a uma nova maneira de lidar com estes dados chama-se *Big Data*. Existem já algumas soluções disponíveis nesta área (o *Apache Hadoop*, o *NoSQL*, e armazenamento em *Massively Parallel Processing* por exemplo), já utilizadas por grandes empresas como a IBM ou a HP.

Exemplos de tecnologias que tornam real a necessidade de conseguir adquirir, adaptar, analisar, armazenar, tratar e aceder a enormes quantidades de dados são:

- **Grande Colisionador de Hadrões** (*LHC – Large Hadron Colider*) do CERN (*Organisation européenne pour la recherche nucléaire*): em 2012 produzia cerca de 25 Peta-Bytes de dados por ano, e possui uma *grid* (ou rede) mundial de mais de 170 centros computacionais para conseguir processar todos os dados produzidos pelo Colisionador;
- **Sistemas Financeiros**: várias organizações utilizam análise de dados de interações dos seus utilizadores para conseguir estabelecer padrões e segmentos. Toda a informação extraída é depois aplicada no desenvolvimento de novas soluções e ofertas aos utilizadores;
- **Instituições Hospitalares**: além de conseguirem agregar vários dados sobre pacientes, incluindo históricos e exames realizados, instituições clínicas tendem a procurar análise de dados para ser possível prever situações para as quais os hospitais devem estar preparados (por exemplo épocas de maior afluência);
- **Indústria Automóvel**: com a incorporação de computadores no automóveis capazes de recolher informações sobre a utilização e desempenho dos mesmos, a indústria automóvel espera agregar dados que permita melhorar o processo de desenho e construção de veículos. A Ford anunciou este ano que já está a utilizar ferramentas de Big Data para analisar os seus dados [22].

Como estes variados exemplos, existem muitos mais [23], que demonstram que é real a demanda por novas tecnologias como estas, que permitem o manuseamento de novas quantidades e tipos de dados.

A aplicação desta tecnologia no projeto prende-se com a preparação do mesmo para receção de grandes quantidades de informação, proveniente de diferentes fontes (por exemplo conteúdo multimédia de exames biomédicos, como Raios X, TAC's, Ecografia, com outros

conteúdos de exames biomédicos, ECG, Oximetria de Pulso, Temperatura; e também informação médica como o historial clínico), algo que se enquadra nas características do *Big Data*. Ao invés de utilizar uma Base de Dados Relacional, pretende-se guardar todos os dados em ficheiros que contém toda a informação relativa aos dados recebidos. No entanto, devido às restrições temporais do projeto, e ao facto de outros desenvolvimentos serem prioritários, foi apenas possível efetuar uma aproximação a esta tecnologia, aplicando a escrita de dados de exames para ficheiros, e lendo os mesmos.

Não será discutido o Estado da Arte em relação ao *Big Data*, devido a não ter sido utilizada tecnologia avançada (por exemplo plataformas como o *Hadoop* ou *HBase*). No entanto, é importante referir que *Big Data* e os conceitos associados foram tidos em conta no desenvolver do projeto.

3 Estado da Arte

Neste capítulo será apresentada a informação relativa a produtos e recursos tecnológicos existentes e passíveis fazerem parte do trabalho a desenvolver no âmbito do projeto. Será dado especial enfâse ao produto OneCare, solução que se pretende melhorar adicionando novas funcionalidades, bem como aos produtos utilizados.

3.1 OneCare

O OneCare é uma linha de produtos da Intellicare, spin-off da ISA (Intelligent Sensing Anywhere, S.A.), empresa sediada em Coimbra e cuja especialidade é a telemetria – o envio de dados a partir de uma localização remota para outra, utilizando sensores [6]. A ISA possui vários anos de experiência no mercado, apostando na investigação e desenvolvimento de novas tecnologias, nas áreas da energia, combustíveis e ambiente. A Intellicare, utilizando o *know-how* da ISA, desenvolve soluções na área da saúde, de modo a proporcionar uma maior qualidade de vida aos utilizadores dos seus produtos. Inserido neste conceito surge o OneCare.

O OneCare tem como principal objetivo a transmissão de dados biomédicos de forma remota, permitindo o acesso a estes a partir de qualquer localização com acesso à Internet. Deste modo, qualquer pessoa com produtos OneCare pode ser monitorizada remotamente, o que se reflete ser de extrema utilidade nos dias de hoje, uma vez que é proporcionado um acompanhamento de situações que necessitam de cuidados continuados comodamente, em ambiente domiciliário, tanto para o utilizador do aparelho como para o prestador de cuidados (seja este um familiar, um médico, etc.). Entre as situações em que o produto OneCare é uma mais valia contam-se as seguintes:

- Pós-operatório que necessite de monitorização de sinais vitais, mas que dispense internamento. Utilizando um produto OneCare, um paciente sujeito a uma operação com um período pós-operatório que necessite de um controlo de sinais vitais pode, em caso de decisão médica, não ficar internado e assim abrir vaga na unidade clínica em questão, ao mesmo tempo que pode encontrar uma situação mais cómoda para recuperar;
- Doentes que necessitem de cuidados especiais e continuados. Os produtos OneCare podem ser utilizados para garantir a segurança de indivíduos em situações especiais, como é o caso de doenças do foro neurológico (exemplo: um individuo que sofra de Doença de Alzheimer, que utilize um produto OneCare pode ser localizado em caso de deteção de anormalidades nos sinais biomédicos monitorizados, ou em caso de alarme ativado pelo próprio individuo);
- Idosos ou indivíduos de saúde frágil podem ver os seus sinais biomédicos monitorizados permitindo uma constante avaliação do estado de saúde do utilizador do aparelho.

Como estes casos podem ser encontrados outros: desportistas que queiram monitorizar a sua performance, indivíduos que queiram monitorizar a variação de sinais biomédicos consoante as suas atividades, são apenas mais dois exemplos.

Para enfrentar este nicho de mercado, a linha de produtos OneCare possui neste momento dois produtos: o *OneCare Sensing* e o *OneCare Safe*.

3.1.1 OneCare Sensing

O *OneCare Sensing* é um *kit* de fácil utilização que permite monitorizar a tensão arterial, frequência cardíaca, peso e glicemia, no domicílio do utilizador. Oferece ainda a possibilidade de um prestador de cuidados ou profissionais de saúde acompanharem o estado de saúde do utilizador, à distância, contactando-o sempre que ocorram alterações relevantes nos

parâmetros avaliados [24]. Possibilita assim um acompanhamento contínuo do estado de saúde do utente permitindo a deteção precoce de situações de risco, de modo a permitir uma intervenção atempada e eficaz em caso de necessidade.

As medições de tensão arterial, glicemia ou peso são feitas pelo utilizador, no conforto da sua casa, ficando automaticamente disponíveis no portal OneCare, na Internet, para poderem ser consultadas pelo utilizador, seus familiares e/ou prestador de cuidados, consoante desejado [24]. A frequência das monitorizações efetuadas pelo utente é ajustada a cada caso, dependendo do tipo de utilizador. As avaliações são feitas através de vários aparelhos sem fios: um medidor de tensão arterial, uma balança ou qualquer outro aparelho de medição de bio-sinais com capacidade de comunicação Bluetooth.

O sistema possui uma funcionalidade importante que consiste na emissão de alertas quando ocorrem desvios nos parâmetros medidos, passíveis de serem validados pelos prestadores de cuidados. Os desvios nos parâmetros medidos são identificados e registados no portal OneCare, gerando alertas automáticos que são registados na base de dados. Além de serem registados, os alertas são disponibilizados no portal web e podem ser enviados por *sms* ou *email* para quem desejado e configurado. Deste modo o utilizador pode ser contactado para despistar a razão do alarme e agir em conformidade.

As condições de alerta são previamente definidas pelos prestadores de cuidados e podem incluir vários níveis, para cada utilizador individualmente. Esta gestão automática e personalizável de alarmes permite ao sistema dispensar uma validação diária de todos os dados por parte do provedor de serviço, bastando que este se preocupe em despistar os alarmes diários emitidos pelo sistema, devidamente categorizados por gravidade da situação. O sistema permite assim um aumento considerável da eficiência de monitorização, sem que isso acarrete uma escalada nos gastos com pessoal especializado em cuidados de saúde.

A seguinte lista resume os benefícios do sistema OneCare Home:

- O seu utilizador pode acompanhar de modo permanente, a sua condição de saúde;
- O utente pode sentir maior confiança e tranquilidade em relação aos seus parâmetros de saúde;
- O utilizador tem acesso fácil à sua informação de saúde atual, com acesso permanente a histórico de dados para comparação com valores anteriores;
- Acréscimo de qualidade de vida, quer do utilizador que vê a sua autonomia acrescida e o acesso aos cuidados de saúde facilitado, quer dos prestadores de cuidados que podem assim monitorizar constantemente bio-sinais do seu utilizador;

3.1.2 OneCare Safe

O *OneCare Safe* é um equipamento de fácil utilização que permite que aos prestadores de cuidados ou familiares acompanharem o bem-estar dos utentes à distância [24]. Esta solução é especialmente indicada no acompanhamento de idosos e de pessoas com necessidades especiais, em ambiente domiciliário. Para isso é utilizado um pequeno comando que cabe na palma da mão e no bolso que possui as seguintes funcionalidades:

- Botão de alerta, passível de ser utilizado no domicílio ou no exterior e que ativa um pedido de auxílio;
- Deteção automática de quedas através de sensor;
- Localização por GPS;
- Chamadas de voz;
- Sistema de alta voz;

- Envio e receção de alertas via SMS e/ou mensagem de voz para familiares ou profissionais de saúde;
- Prova de vida;

Figura 9: One Care Safe [24]

O OneCare Safe permite assim que o utente esteja sempre acompanhado, dentro e fora de casa. Devido às suas características o equipamento pode ser transportado em várias localizações, embora tal condicione a aplicabilidade do detetor de quedas, sendo que utilizar o dispositivo como pendente ao pescoço é uma opção que otimiza esta funcionalidade.

Este produto apresenta-se assim como uma solução viável com os seguintes benefícios:

- O utente pode ter ajuda dos prestadores de cuidados ou familiares sempre que necessitar, nomeadamente em caso de emergência ou especificamente sofrer uma queda;
- Os prestadores de cuidados ou familiares podem acompanhar o seu bem-estar, resultando num benefício mútuo;
- Maior confiança e tranquilidade do utente, mesmo habitando sozinho;
- Maior qualidade de vida, com autonomia acrescida e acesso facilitado ao auxílio, sempre que necessitar;
- O utente pode permanecer no conforto do seu domicílio se assim o desejar, estando sempre *acompanhado*;
- O utente vê uma maior garantia de independência nas tarefas diárias, mesmo aquelas que envolvem a saída da sua habitação.

3.1.3 Considerações Globais

Com estes dois produtos, a gama OneCare visa garantir um aumento de qualidade de vida aos seus utilizadores. Consequência também da sua utilização é uma alteração das necessidades do próprio utilizador, e logo, dos seus próximos. Os benefícios para os responsáveis pelo cuidado daqueles dos utilizadores de produtos OneCare podem resumir-se na seguinte lista:

- Melhoria da qualidade do serviço prestado, em caso de ser um prestador de cuidados ou entidade médica;
- Visualização do histórico das monitorizações dos utentes;
- Receção de alarmes aquando da ocorrência de desvios do estado de saúde e bem-estar;
- Diagnóstico precoce e prevenção de situações de risco;

- Gestão mais eficaz da prestação de cuidados, sendo mais alternativa aos métodos tradicionais;

3.2 OneCare Sensing Plus – Contexto do Projeto

Da análise do OneCare surge a identificação de oportunidades de melhoria do sistema, que permitam ampliar o leque de opções técnicas e práticas do produto.

Entre essas oportunidades, uma das mais óbvias é o acrescentar de sensores ao pacote, para tornar a monitorização de parâmetros biomédicos mais completa e eficaz.

Outra questão, menos óbvia mas não menos importante, é a modernização dos protocolos e sistemas de comunicação utilizados para a transmissão de dados entre o utilizador e o sistema interno da ISA. Esta comunicação é feita utilizando servidores da ISA capazes de receber e armazenar mensagens e respetivos dados. A fonte de dados (por exemplo um *tablet* ligado à Internet) envia os dados para um servidor pré-definido aquando da conceção do produto. O servidor tem programas que tratam da sua receção, tratamento e armazenamento adequado. O MQTT, um protocolo de comunicação diferente dos utilizados atualmente, mais rápido e eficaz, será assim implementado.

O modo como os dados são armazenados também é outro dos pontos suscetíveis de análise e possível melhoria. Atualmente é utilizado um sistema convencional de Base de Dados Relacional, mas a constante evolução dos sensores torna a quantidade de dados enviada cada vez maior, o que pode requerer uma nova abordagem. O conceito de *Big Data* é assim explorado neste projeto, como forma de aumentar a capacidade eficaz de armazenamento sem perder qualidade de organização e abrindo portas a um leque maior de abordagens futuras à forma como os dados são guardados e acedidos.

Espera-se assim melhorar a solução existente, adicionando funcionalidades (novos sensores), melhorando o sistema (novo protocolo de comunicação e novo método de armazenamento de dados) além de proporcionar uma janela aberta à evolução do sistema OneCare já que tanto o MQTT como o *Big Data* são duas tecnologias emergentes na tecnologia atual.

3.3 Produtos de Monitorização Remota

Existem já no mercado, especialmente nos Estados Unidos da América, vários produtos que permitem a um utilizador comum uma constante monitorização dos seus sinais vitais. Dentro do leque de produtos diferentes, os mecanismos de aquisição e de posterior acesso aos dados variam bastante, pelo que serão enumerados alguns destes produtos para se compreender melhor o nicho de mercado que o produto OneCare visa atingir:

- **Toumaz SensiumVitals**: sistema de monitorização de sinais vitais de pacientes de cuidados gerais. O sistema é composto por sensores do tipo penso, leves, que monitorizam a frequência cardíaca, taxa respiratória e temperatura a cada dois minutos, enviando estes dados de forma sem fios, e através de um dispositivo intermédio, para um centro de cuidados ou outros dispositivo com acesso ao serviço. Foi desenvolvido tendo em vista o uso hospitalar [25].
- **Corventis AVIVO Mobile Patient Management (MPM) System**: utilizando este produto, os pacientes são monitorizados continuamente através do dispositivo PiiX. O PiiX tem a forma de um penso e coloca-se na pele, na zona do peito. É resistente à água e isso permite que não seja removido (por exemplo aquando do banho). Coleta automaticamente informação relativa à frequência cardíaca, variabilidade de frequência cardíaca, taxa respiratória, postura, atividade e ECG (ativado quando são detetadas anormalidades rítmicas), e por ter pequenas dimensões, pode ser utilizado

por baixo da roupa e assim *seguir* o paciente na sua rotina diária. A informação é enviada através de tecnologia sem fios do PiiX para outro aparelho de pequenas dimensões, o zLink que por sua vez envia os dados para a Corventis. Na Corventis, técnicos de saúde analisam os dados recebidos e em caso de serem válidos, disponibilizam-nos para acesso de médicos ou outro utilizador indicado [26].

- **iRythm Zio Patch**: o *Zio Patch* é um dispositivo de pequenas dimensões que possui dois elétrodos e um processador, bem como memória interna. O aparelho deve ser colocado no peito do utilizador e coleta dados de eletrocardiograma durante até 14 dias ininterruptamente. É resistente à água e por isso não deve ser retirado durante a monitorização. No final, o aparelho é removido, enviado para os laboratórios da *iRythm*, e os dados recolhidos são analisados, sendo posteriormente comunicados ao utilizador [27].

Tal como estes produtos, um trabalho de pesquisa permite acesso a dezenas de soluções existentes no mercado, todas baseadas nos mesmos princípios. O mercado Norte-Americano é sem dúvida prolífico neste tipo de soluções, embora já existam produtos na Europa com os mesmos objetivos.

3.4 Produtos utilizados no Projeto

Ao preparar este projeto, pretende-se recorrer ao uso de produtos modernos e atuais, garantido uma solução funcional, viável, e tecnologicamente acessível. Para tal foram utilizados os recursos que o produto OneCare disponibiliza, ajustados às necessidades do projeto. De seguida são apresentados os produtos que foram utilizados para recolher dados de sinais biomédicos.

3.4.1 Plux BioSignals

Para recolher sinais de Eletrocardiograma, Temperatura e Taxa Respiratória foi utilizado o Plux BioSignals [28]. Este é um produto desenvolvido pela Plux, capaz de monitorizar diversos parâmetros biomédicos simultaneamente, através de vários sensores conectados via fios à mesma unidade central, uma pequena caixa do tamanho de um telemóvel. Os sensores incluídos são:

- **Eletroencefalografia (EEG)**: O sensor de EEG incluído no produto permite não só efetuar uma monitorização clássica de EEG como também medir apenas áreas mais reduzidas. Com uma configuração de três elétrodos, três superfícies são utilizadas para detetar potenciais elétricos em regiões específicas do escalpe, em relação a um elétrodo de referência que deve ser colocado numa zona de baixa atividade muscular. O resultado final é a diferença amplificada entre os três sinais, o que reduz o ruído indesejado. Este sinal permite inferir sobre a atividade cerebral, pois esta envolve a transferência de informação entre neurónios sob a forma de potenciais elétricos. Medições em tempo-real e localizadas da atividade elétrica cerebral à superfície do escalpe podem ter várias aplicações médicas, como em casos de epilepsia, no estudo dos estados do sono, interfaces neuronais para computação, entre outras.
- **Pressão de Volume Sanguíneo (BVP** do inglês *Blood Volume Pressure*)**:** O sensor de Pressão de Volume Sanguíneo é um sensor ótico não-invasivo que mede variações do volume sanguíneo numa extremidade arterial, baseado na técnica de fotopletismografia. Este sensor possui uma sonda para ser colocada na ponta do dedo com uma fonte de luz vermelha e um fotodetetor. Estes dois componentes estão em modo de deteção de transmissão, e devido à sua configuração permitem assinalar as duas fases do ciclo cardíaco (sístole e diástole). A aplicação mais comum deste tipo de sensor é a medição da frequência cardíaca e da variabilidade de frequência cardíaca. No entanto, pode ser utilizado para outro tipo de estudos, como a avaliação da

resistência arterial, da elasticidade da aorta ou até para obter o valor de pressão arterial.

- **Eletromiografia (EMG):** O movimento muscular envolve a ação de músculos e nervos, pelo que existe uma corrente elétrica envolvida. Medindo a atividade elétrica nos músculos e nervos associados pode portanto ajudar a detetar doenças musculares, que levem a uma condição anormal dos mesmos (fraqueza ou paralisia por exemplo). Este sensor é capaz de efetuar eletromiografias com elétrodos superficiais bipolares – Eletromiografia de Superfície (*sEMG*) - monitorizando a ativação muscular, ao contrário de uma eletromiografia intramuscular em que apenas algumas fibras são estudadas. A sEMG é utilizada para avaliar e gravar a atividade elétrica associada ao movimento muscular com variadas aplicações clínicas e biomédicas, em áreas como a neurologia, reabilitação, ortopedia, ergonomia, desporto, etc. Um exemplo é o caso da fisioterapia: utilizando esta técnica, a atividade muscular é monitorizada e o paciente recebe um estímulo (visual ou auditivo por exemplo) para o ajudar a saber quando está a ativar o músculo – *BioFeedback*. A sEMG permite a um clínico identificar doenças neuromusculares, através da avaliação de magnitude e duração do impulso elétrico. Outra aplicação biomédica importante é a utilização dos impulsos elétricos detetados para controlar próteses ou outros dispositivos com uma interface adequada (por exemplo um telemóvel).
- **Atividade Eletrodérmica (AED** ou **EDA** do inglês *Electrodermal Skin Activity*): Pode ser definida como uma variação transiente de certas propriedades elétricas da pele, associadas com a atividade das glândulas sudoríparas, provocada por um estímulo que leve a uma resposta da pele. O sensor de AED mede a atividade da pele com grande de sensibilidade, com um sinal de ruído baixo e pouca amplificação. Deste modo é possível detetar pequenas variações desta propriedade da pele com muita precisão. Algumas aplicações deste sensor são a deteção de estados de concentração, cognição e emoção, ou como ferramenta auxiliar numa cirurgia de Endoscopia Torácica.
- **Acelerómetro**: O acelerómetro triaxial é baseado na tecnologia *MEMS* (*Micro Electro-Mechanical Systems*) e foi desenvolvido para aplicações biomédicas em que monitorizações cinemáticas e de movimentos sejam necessárias. Este sensor tem a capacidade de medir acelerações relativas a quedas livres, bem como calcular a magnitude e direção dessas acelerações como uma quantidade vetorial. O vetor resultante pode ser utilizado para calcular uma posição, vibração, choque, queda, etc. Todas as conclusões que se podem tirar deste sensor dependem da posição em que está acoplado.
- **Taxa Respiratória:** Os sinais respiratórios obtidos estão ligados direta ou indiretamente aos volumes pulmonares aquando de cada ciclo respiratório. Medições indiretas da respiração podem ser feitas utilizando este sensor, que incorpora um sensor piezoelétrico (*Piezo Film Technology*). O sensor mede as variações de comprimento relacionadas com movimentos abdominais e torácicos, obtendo um sinal respiratório de alta sensitividade e baixo ruído, onde os ciclos respiratórios podem ser observados. Algumas aplicações incluem a medição da deslocação da cavidade torácica, além da óbvia observação do ciclo respiratório. O diagnóstico de doenças do sono (como a apneia do sono, caracterizada por pausas na respiração ou numa respiração anormalmente reduzida) é outra aplicação. A monitorização da respiração em atletas em exercício para determinar os níveis de ventilação, e relacionar estes dados com a sua performance é também uma possível utilização desta técnica.
- **Eletrocardiograma (ECG) –** A condução de potenciais de ação através do coração gera correntes elétricas que podem ser detetadas por elétrodos colocados na pele. O registo de variações elétricas que acompanham a atividade do coração constitui um ECG. A análise da variação da forma e duração das ondas constituintes do ECG é uma das

ferramentas mais utilizadas no diagnóstico do estado funcional do coração. Um trio de elétrodos para ECG de baixo ruído são especialmente desenhados para um posicionamento local e permitem um sinal de aquisição de boa qualidade, maximizando a performance do sensor e providenciando um sinal de alta resolução para análise fina.

- **Temperatura** – Sensor de temperatura, que visa detetar a temperatura corporal do utilizador.

Figura 10: Aparelho Plux semelhante ao utilizado

Destes sensores foram utilizados o de Temperatura, Eletrocardiograma (que também calcula Frequência Cardíaca) e Taxa Respiratória. Ao mesmo tempo que adquire os valores, o produto tem a capacidade de os enviar via Bluetooth para outro dispositivo que possua comunicações Bluetooth (um PC, um smartphone ou um *tablet* por exemplo). O produto utilizado corresponde à versão anterior à atual, que sofreu entretanto algumas modificações quer a nível visual quer de funcionamento.

3.4.2 Nonin 4100

Figura 11: Nonin 4100

O Nonin 4100 é um Oxímetro de Pulso, capaz de calcular com precisão a saturação percentual de Oxigénio no sangue (SpO_2), um indicador importante do estado se saúde do paciente. O aparelho envia estes dados para o dispositivo coletor de dados através de tecnologia Bluetooth. Além destas funções, consegue também calcular a frequência cardíaca do utilizador, através da técnica de fotopletismografia. O funcionamento deste aparelho baseia-se nos princípios enunciados na secção 2.1.4, sendo que os comprimentos de onda deste dipositivo são de 660 nanómetros para a luz vermelha e de 910 nanómetros para a luz infravermelha [29]. O dispositivo coloca-se na ponta do dedo, como um dedal, e através da deteção da quantidade de luz transmitida através dos tecidos do dedo, deteta a saturação de Oxigénio no sangue arterial em circulação, bem como a frequência cardíaca do utilizador.

Ao mesmo tempo que está a monitorizar os dois parâmetros referidos, o aparelho envia os dados via Bluetooth para outro dispositivo que comunique com este, permitindo assim o acesso em tempo real aos dados adquiridos.

3.5 Java

O Java é atualmente tecnologia propriedade da Oracle, constituindo uma das linguagens de programação mais utilizadas no mundo. As suas aplicações vão desde o desenvolvimento de jogos e aplicações móveis até à criação de *software* empresarial, estando presente no dia-a-dia de cada vez mais pessoas com acesso à tecnologia. Foi esta a linguagem escolhida para criação do programa que receberá dados biomédicos, e de ferramentas para ler esses mesmos dados, permitindo retirar conclusões acerca dos mesmos de uma perspetiva médica.

São necessários diversos elementos ao desenvolvimento nesta linguagem. Estes estão agrupados num único pacote que visa facilitar o seu acesso: o JDK (*Java Development Kit*, inglês para Kit de Desenvolvimento em Java), que constitui um conjunto de utilidades para iniciar o desenvolvimento em Java. O JDK é um caso particular de SDK (*Software Development Kit*, inglês para Kit de Desenvolvimento de Software) que como o nome indica é um conjunto de ferramentas que visa o permitir o desenvolvimento de Software. A versão atual do JDK, bem como das principais ferramentas do Java é a 7, tendo saído já 40 atualizações a esta versão [30]. Segundo o website oficial do Java, os constituintes do JDK são:

- **Ferramentas de Desenvolvimento**: ferramentas que constituem a base que permite ao utilizador desenvolver, executar e corrigir programas escritos em Java;
- ***Runtime Environment*:** embora possa ser descarregado em separado, o JRE (*Java Runtime Environment*) vem incluído no JDK, e constitui a ferramenta fundamental que permite correr todos os programas escritos em Java (quer externos, quer os desenvolvidos pelo utilizador). Inclui livrarias e outros ficheiros que permitem a execução de programas Java;
- **Outras Ferramentas:** ferramentas adicionais que podem ser utilizadas, como por exemplo **Livrarias Adicionais** (ferramentas extra e documentação respetiva, que podem ser utilizadas no desenvolvimento em Java), **Java DB (**a distribuição da Oracle do *Apache Derby*, uma Base de Dados *open source* relacional implementada exclusivamente em Java), funções de compatibilidade com programação em código nativo (C) e ferramentas para uso com ***JavaFX*** (plataforma para desenvolvimento de Aplicações Ricas de Internet, aplicações Web que possuem muitas características de aplicações tradicionais).

Fazendo *download* do JDK, é possível iniciar de imediato o desenvolvimento de programas em Java. No entanto, é frequente utilizar uma IDE que proporciona uma interface gráfica agradável à programação, bem como uma configuração facilitada de ferramentas incluídas. É assim mais simples e intuitivo programar utilizando um destes programas. O IDE escolhido foi o que mais se adequou às necessidades do projeto, tendo em conta as suas funcionalidades, a sua adequabilidade à tecnologia a ser explorada, bem como a compatibilidade com os recursos já existentes do OneCare a ser utilizados.

3.5.1 Eclipse

O IDE que foi escolhido foi o Eclipse, tendo sido utilizada a versão 4.2 (*juno*) que era a mais atual à data do início do desenvolvimento do programa. Surgiu recentemente a versão 4.3 – em Junho de 2013 mas uma vez que nessa altura o projeto já estava em fase de finalização, não foi utilizada esta versão. O Eclipse é um IDE maioritariamente escrito em Java que permite a criação e edição de código escrito em várias linguagens, entre elas o Java, C, C++, JavaScript,

entre outras. Permite a incorporação de *plug-ins* que aumentam as capacidades e funcionalidades do Eclipse, tornando-o personalizável e mais versátil.

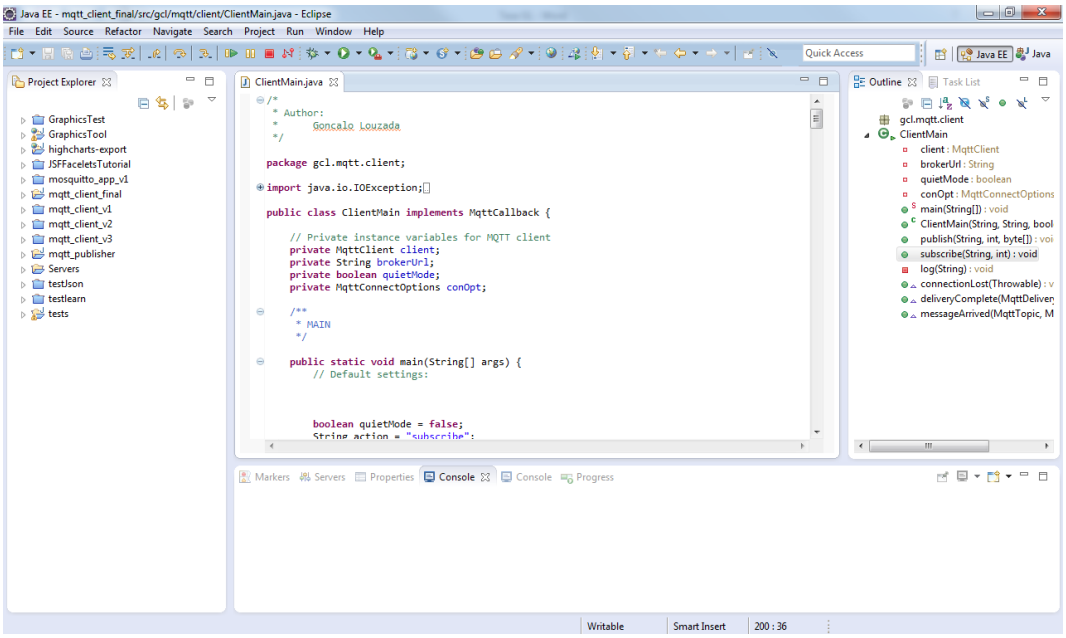


Figura 12: Ecrã principal do Eclipse

Existem vários pacotes disponíveis para *download*, pelo que o escolhido foi o que incorpora o desenvolvimento de Java EE (ou JEE - *Java Enterprise Edition)* que permite programação dirigida a servidores, algo que será utilizado neste projeto. O pacote inclui, além do Eclipse, ferramentas que de outra maneira teriam de ser configuradas manualmente, mas que assim são automaticamente incorporadas no Eclipse e deste modo facilitam a sua utilização.

3.5.2 JSF – *JavaServer Faces*

Além do Eclipse, outra utilidade Java utilizada foi o JSF – *JavaServer Faces*, uma ferramenta Java que se insere no desenvolvimento de Aplicações Web e na criação de interfaces com o utilizador. A sua versão estável mais recente é a 2.1 [31] e foi a utilizada no projeto para desenvolvimento de uma interface com o utilizador, para este selecionar e visualizar dados biomédicos.

3.6 MQTT – Message Queuing Telemetry Transport

O MQTT foi a tecnologia escolhida para desempenhar as funções de comunicação entre as fontes de dados biomédicos e o programa final, que irá receber e armazenar estes dados. Insere-se no conceito de *Internet of Things*, que se refere a objetos identificáveis univocamente e a sua representação virtual numa estrutura do tipo Internet. A *Internet of Things* é uma tecnologia em grande crescimento, existindo um estudo da ABI Research que estima que em 2020 existirão 30 mil milhões de dispositivos conectados através desta tecnologia [32].

Segundo o *website* oficial do MQTT, a versão mais recente do MQTT é a 3.1 [33]. Esta versão do MQTT inclui as seguintes especificações oficiais [34]:

- O transporte da mensagem é agnóstico do conteúdo da mesma;
- A conectividade em rede é atingida utilizando o protocolo TCP/IP;
- A entrega de mensagens tem três Qualidades de Serviço (*QoS - Quality of Service*):
    1. Uma vez (*At most once*), *QoS* igual a **0**. As mensagens são entregues consoante as condições de rede assim o permitam. Podem ocorrer perdas de mensagens

ou duplicações, sendo este o nível mais baixo de garantia. Este QoS pode ser utilizado por exemplo com sensores de ambiente (por exemplo temperatura ou humidade), uma situação em que uma perda de informação não tem consequências graves;

2. Pelo menos uma vez (*At least once*), QoS igual a **1**. Todas as mensagens são entregues pelo menos uma vez, existindo garantia de entrega. No entanto podem continuar a ocorrer duplicados;
3. Exatamente uma vez (*Exactly once*), QoS igual a **2**. Todas a mensagens são entregues uma e uma só vez aos subscritores. Este nível pode ser utilizado por exemplo em sistemas de pagamento em que é necessário garantir que os dados são processados corretamente e a margem de erro é mínima;

- O *overhead*[3] de transporte é pequeno (o cabeçalho de tamanho fixo possui apenas 2 bytes), e as trocas de protocolo são minimizados, para reduzir o trafego em rede;
- Mecanismo de notificação de partes interessadas em caso de desconexão anormal de um cliente, utilizando a funcionalidade de *Last Will and Testament*. Basicamente, em caso de desconexão inesperada, o cliente em causa envia uma mensagem automática.

Existem já vários casos de utilização bem-sucedida do MQTT em projetos de dimensão e exigência elevadas, entre eles:

- Aplicação *Facebook Messenger* para iOS, *Android*, e Windows: Utilizado por ser um protocolo rápido e com pouca utilização de bateria e largura de banda [35];
- Comunicação remota de *pacemakers* com a unidade clínica de St. Jude nos Estados Unidos. Os *pacemakers* enviam dados remotamente para os clínicos possam monitorizar o seu estado [36];
- Monitorização e controlo de *pipelines* de petróleo [36];
- *Habitações inteligentes*, ou seja, casas que ligadas à Internet, permitem ligar e desligar dispositivos, e monitorizar o seu funcionamento [36].

Além destes exemplos, um trabalho de pesquisa permite encontrar muitos mais que, por uma questão de sentido prático deste documento, não serão aqui discutidos.

Convém assim compreender qual o estado atual da tecnologia associada ao MQTT, quais os *brokers*, quais os clientes e quais as linguagens de programação que já possuem desenvolvimentos nesta área.

A primeira análise feita foi a *brokers* existentes atualmente e que pudessem ser utilizados no decorrer do projeto:

- **Apache Active MQ**: *broker* de mensagens com suporte a vários protocolos de comunicação (entre eles o MQTT) e a clientes escritos em várias linguagens de programação. Além destas funcionalidades, é um projeto *open source* desenvolvido em comunidade, parte da Apache Foundation [37];
- ***Mosquitto***: *broker* de mensagens que implementa a versão 3.1 do MQTT. Projeto *open source* e que se define como simples e de fácil configuração. Suporta ligações seguras (SSL - *Secure Sockets Layer*, um protocolo criptográfico de comunicação segura através da Internet [38]) [18].

[3] *Overhead* é qualquer processamento ou armazenamento em excesso, seja de tempo de computação, de memória, de largura de banda ou qualquer outro recurso que seja requerido para ser utilizado ou gasto para executar uma determinada tarefa [52].

Além destes *brokers*, entre os quais foi necessário decidir, existem outros produtos com *brokers* que suportam MQTT. Uma lista de *software* passível de ser utilizado está disponível no site oficial do MQTT [33]. Entre os *brokers* disponíveis mas não considerados encontram-se os seguintes:

- **Xively**: produto que aplica o conceito de *Cloud Computing* (computação na nuvem) à *Internet of Things*. Possui várias API's (*Aplication Programming Interface*) que permitem a incorporação de várias tecnologias, entre elas o MQTT. Trata-se de um produto comercial, não sendo na verdade um *broker* de mensagens MQTT genérico mas sim uma implementação do conceito [33] [39].
- **Hive MQ**: *broker* de MQTT desenvolvido especialmente a pensar em empresas que procuram incorporar tecnologias M2M na sua rede de produtos. É compatível com vários Sistemas Operativos e suporta SSL. Trata-se de *software* comercial, com uma versão gratuita para uso privado, mas devido à natureza do projeto (parceria empresarial), tal torna-se numa desvantagem [40].
- **IBM Integration Bus (antigo Websphere Message Broker)**: *broker* integrado da família de produtos comerciais *WebSphere*, da IBM. Ferramenta muito completa, de cariz empresarial e que permite a utilização de diversos protocolos de comunicação [41], mas que está inserida num conjunto de ferramentas muito mais amplo e que não vai de encontro áquilo que é utilizado atualmente pela ISA nos seus produtos. Além disso, o custo desta ferramenta não justificava a utilização que lhe seria dada, sendo possível com *brokers* mais simples, leves e *open source* atingir os objetivos propostos.
- **IBM Lotus Expeditor micro *broker*:** *broker* integrado no produto IBM Lotus Expeditor, um conjunto de ferramentas para desenvolvimento de aplicações. O objetivo deste *software* é desenvolver aplicações que correm num cliente local mas que apresentam a mesma facilidade de gestão e manutenção de aplicações baseadas em *Web* [42].
- **RSMB - Really Small Message Broker:** *broker* de mensagens MQTT, produzido pela IBM e distribuído gratuitamente (mas não *open source*) para uso pessoal ou testes. Adequado para desenvolvimento, sistemas embebidos, ou utilizações em sistemas de pequena a média dimensão. Possui uma API para um cliente em escrito em C [33].
- **Moquette:** *broker* ainda em fase inicial de desenvolvimento, com o objetivo de ser um *broker* de mensagens MQTT e possuir uma livraria para cliente [43]. Por ser um projeto ainda pouco desenvolvido e pouco sólido, não foi tido em conta para o projeto.
- **Rabbit MQ:** *broker open source* com suporte para MQTT. Pode ser utilizado na maioria de sistemas operativos principais, e pelo facto de ser *open source* já possui ferramentas em várias linguagens (Java incluído) [44].

Dada a grande expansão que esta tecnologia tem experienciado, é normal que mais *brokers* estejam disponíveis cada vez mais frequentemente, ou que os atuais sofram evoluções. No entanto, procurou-se ter uma visão global da maioria de soluções existentes no mercado.

Tendo em conta as propriedades, funcionalidade e custos dos *brokers* analisados, a escolha recaiu apenas sobre o *Apache Active MQ* e o *Mosquitto*. Os dois principais fatores para esta seleção foram:

- A maioria dos *brokers* é de cariz comercial, necessitando de licenças (muitas vezes de custos elevados) para uso empresarial. Ora dado que o projeto se insere num produto empresarial, seria necessário adquirir licenças para poder prosseguir com uma dessas opções;
- O projeto visa ser uma melhoria ao produto OneCare, sem entrar em conflito com o já existente, ou seja, procurar ao máximo manter a compatibilidade com o que já está desenvolvido.

Ainda que estas são as duas principais razões para que tenham sido escolhidos estes dois *brokers* para decisão final, há que ter ainda em conta as suas funcionalidades relativas ao MQTT:

- Compatibilidade com comunicação segura (SSL);
- Compatibilidade com clientes MQTT escritos em Java;
- Suporte à versão mais recente do MQTT (3.1) e todas as especificações inerentes à mesma (já descritas)

Deste modo, testou-se inicialmente o Active MQ, já que este se encontrava em funcionamento nos servidores da ISA, para utilização com vários produtos (incluindo o OneCare). No entanto, após vários testes verificou-se que este ainda possuía alguns *bugs* no que toca à sua utilização como *broker* de MQTT. O principal, e que foi detetado, foi que quando um cliente MQTT se desconectava do mesmo, o *broker* mantinha a identificação do cliente como se este estivesse conectado durante algum tempo. Tal poderia levar a erros na entrega de mensagens e fazia com que fosse necessário esperar alguns minutos até se conseguir nova conexão através do mesmo cliente. Uma vez excluído uso do Active MQ (decisão tomada em concordância com a opinião de membros da ISA responsáveis por este produto e tecnologia) foi testado o uso do *broker* Mosquitto, instalando-o nos servidores da ISA. Após vários testes (discutidos no Capítulo 6) esta verificou-se ser uma escolha acertada.

Escolhido o *broker*, foi necessário decidir qual a ferramenta que permitiria desenvolver o cliente MQTT que será responsável pela comunicação com outros dispositivos, e receber dados. Deste modo, procuraram-se soluções existentes que fossem desenhadas a apontar a uma utilização em Java, e que incorporassem as mais recentes funcionalidades do MQTT. Foram consideradas duas opções:

- **Eclipse Paho**: este projeto providência acesso a API's que possuem funcionalidades de comunicação para protocolos associados à *Internet of Things*, entre estes o MQTT. O projeto é *open source*, e entre essas API's encontra-se uma para Java, que assim seria adequada à utilização no projeto [45]. Esta API facilitaria o processo de estabelecimento de comunicação a ser utilizada pelo cliente MQTT.
- **Fusesource API**: como o nome indica, API desenvolvida pela *Fusesource* (atualmente adquirida pela *Red Hat*) e que também disponibiliza todas as funcionalidade necessárias para que um cliente MQTT escrito em Java possa comunicar com um *broker*, e assim com outros clientes MQTT. Também é um projeto *open source* [46].

A escolha entre estas duas API's foi baseada na simplicidade de aplicação das mesmas, bem como na quantidade de informação existente sobre ambas, dado que em ambos os casos estão reunidas todas as condições para que se possa estabelecer comunicação MQTT. Sendo que da parte da ISA e da equipa do OneCare nada foi imposto, a escolha foi tomada em direção da utilização da API Paho, que desempenha todas as funções necessárias.

Relativamente ao MQTT, este é o estado atual da tecnologia, sabendo antecipadamente que em todas as áreas, especialmente em tecnologias da informação e comunicação, a evolução tem um ritmo elevado e por isso todos os dias novos passos são dados rumo ao futuro. Procurou-se ser o mais atual e rigoroso possível, sem comprometer a compreensão dos dados encontrados.

As escolhas recaíram assim no *Mosquitto* para *broker* e Eclipse Paho para Java API, sendo que estas decisões tiveram em conta requisitos internos do produto OneCare, o desempenho destas duas opções nos respetivos campos, bem como o privilégio da tecnologia *open source*, por disponibilizar muito apoio comunitário e liberdade operacional total.

4 Arquitetura do Projeto

De modo a atingir os objetivos propostos, foi iniciada a conceção de uma arquitetura de projeto que permita uma orientação eficaz do trabalho, ao mesmo tempo que garante o cumprimento das metas com a performance desejada. A solução encontrada passou por utilizar a tecnologia disponível mais recente num esforço para encontrar um conjunto de ferramentas que se complementem.

4.1 Pré-Considerações

4.1.1 OneCare

Antes da descrição fundamental do projeto é necessário compreender o contexto prático do mesmo. Como apresentado aquando da secção 3.2, o produto final irá inserir-se no produto OneCare, pelo que todo o trabalho desenvolvido tem em vista a integração da solução desenvolvida no OneCare. Um fator óbvio que nasce desta condição é que os recursos tecnológicos a serem utilizados são os que a ISA possui. Assim sendo, e dando seguimento ao contexto já apresentado, a solução passa por desenvolver um programa a ser instalado nos servidores da ISA, capaz de receber dados através do protocolo de comunicação MQTT e que os armazene em ficheiros adequados. Posteriormente existe um segundo programa que permite o acesso aos dados armazenados, visualizando os dados obtidos de forma gráfica e permitindo assim uma melhor compreensão e análise dos mesmos.

4.1.2 Conceito Global do Projeto

O projeto apresentado faz parte de um projeto mais amplo, no qual o trabalho desenvolvido se vai inserir. O conceito global está resumido na Figura 13, que permite distinguir duas partes fundamentais: a aquisição, a receção, armazenamento e leitura de dados. O presente projeto corresponde à segunda parte do conceito global (secção rodeada a verde na Figura 13).

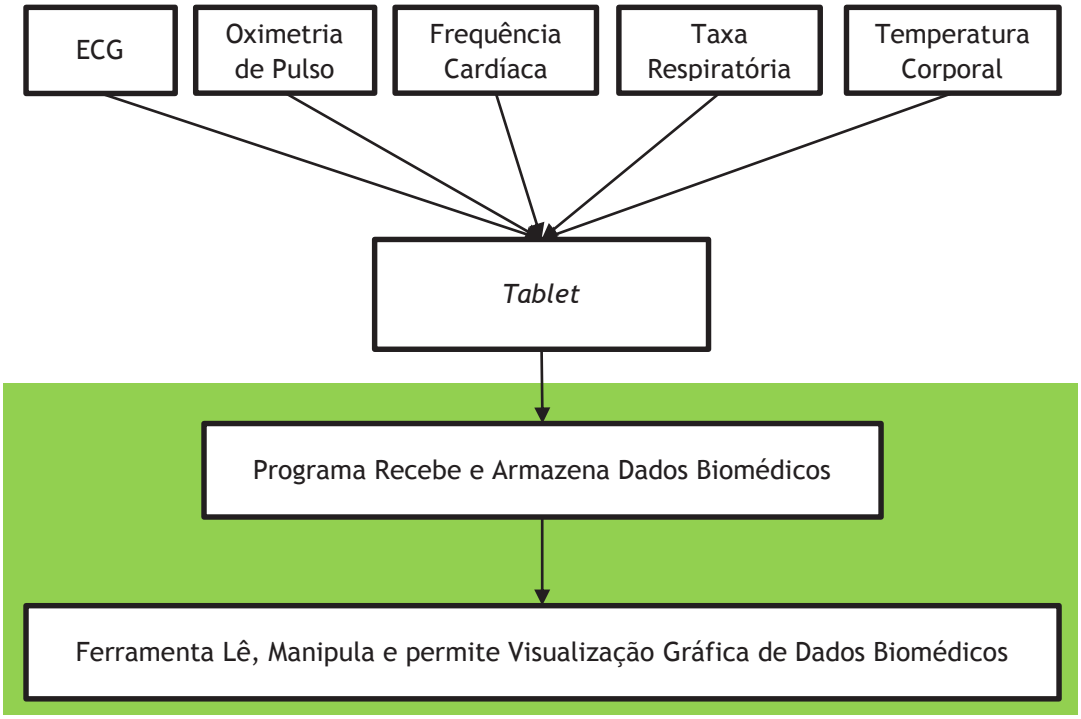


Figura 13: Conceito Global do Projeto

O esquema da Figura 13 pode ser lido do seguinte modo:

1. Os dados de sinais biomédicos (ECG, Oximetria de Pulso, Frequência Cardíaca, Taxa Respiratória ou Temperatura Corporal) são recolhidos;
2. Os sensores ativos enviam os dados recolhidos para um *tablet* com sistema operativo *Android*, através de tecnologia Bluetooth;
3. O *tablet* recebe os dados e envia os mesmos através de Internet para um *broker* instalado num servidor, utilizando MQTT;
4. Um programa conectado ao *broker* referido recebe, descodifica e armazena os dados;
5. Os dados ficam disponíveis para serem lidos, manipulados e visualizados graficamente através de outro programa.

Os sensores utilizados apresentam especificações que condicionam o projeto, nomeadamente a sua taxa de aquisição de dados. Este fator é especialmente importante no que toca à qualidade dos dados e consequentemente na sua visualização.

Dada a complexidade do projeto, foi necessário definir alguns parâmetros transversais ao mesmo, para que todas as partes se complementem e funcionem perfeitamente, sem problemas de compatibilidade. Os pormenores funcionais que definidos em concordância no conceito geral do projeto são:

- Taxa (ou Frequência) de Aquisição de Dados pelos sensores: quando este valor não é pré-definido pelo Sensor propriamente dito, é controlado pelo *tablet;*
- Mensagem: elemento fundamental do projeto, a mensagem enviada pelo *tablet* e recebida pelo programa contém a informação extraída dos sensores. Foi redigida utilizando a tecnologia *JSON*, e o seu conteúdo foi criteriosamente definido.

4.2 Arquitetura Global

O projeto divide-se em duas partes internas fundamentais (Figura 14):

- Um programa, instalado num servidor que possui um *broker* MQTT em funcionamento, recebe dados de sinais biomédicos e armazena-os apropriadamente;
- Uma ferramenta de visualização gráfica de sinais biomédicos permite posterior acesso aos dados recebidos e armazenados.

Figura 14: Partes fundamentais do projeto

4.2.1 Arquitetura Lógica da Primeira Parte

A primeira parte do projeto é um programa que permite a receção contínua de dados enviados para um servidor da ISA. O funcionamento do mesmo pode ser discriminado em três partes que se complementam:

1. O cliente MQTT, que recebe a mensagem;
2. Componente de leitura de dados, que descodifica a mensagem recebida e extrai os dados;
3. Componente de armazenamento e organização de dados, que guarda adequadamente os dados recebidos;

Deste modo visa-se garantir a otimização do funcionamento do programa e o cumprimento do seu propósito.

Figura 15: Arquitetura lógica da primeira parte do projeto

4.2.2 Arquitetura Lógica da Segunda Parte

A segunda parte do projeto é uma ferramenta que permite a visualização gráfica dos dados armazenados pelo programa da primeira parte do projeto.

A ferramenta é composta por uma Página HTML que se serve de um conjunto de elementos desenvolvidos em Java para fazer a representação gráfica dos dados selecionados. Esta ferramenta pode subdividir-se em três partes com os seguintes objetivos:

1. Leitura de ficheiro de dados: é lido um determinado ficheiro e extraída a informação contida no mesmo;
2. Pré-análise de dados: os dados lidos são analisados para garantir que a sua visualização é correta;
3. Visualização gráfica de dados: numa Página HTML, é utilizado um JavaScript para apresentar um gráfico com os dados devidamente representados.

Figura 16: Arquitetura lógica da segunda parte do projeto

5 Metodologia

Neste capítulo serão descritos todos os mecanismos desenvolvidos para atingir os objetivos traçados. O desenvolvimento foi dividido em duas etapas: a criação do programa que recebe e armazena os dados biomédicos (um cliente MQTT que irá subscrever a tópicos do *broker* instalado no servidor da ISA) e a elaboração de uma ferramenta que permite o acesso e visualização gráfica dos dados armazenados pelo programa. O programa foi escrito em Java, enquanto a ferramenta incorpora várias tecnologias, nomeadamente HTML, Java, JSF e JavaScript. O capítulo visa assim apresentar o trabalho produzido, detalhando os métodos utilizados e assim proporcionando a compreensão dos resultados obtidos.

5.1 Programa de Receção de Armazenamento de Dados

Esta é a componente do projeto que será instalada num servidor, responsável por receber todos os dados e armazená-los corretamente.

Para desenvolvimento deste programa foi selecionada a linguagem de programação Java, utilizada no produto OneCare (importante para fácil integração), com aplicabilidade na maior parte das situações e compatibilidade com a utilização de MQTT. Este programa subdivide-se em quatro unidades funcionais, todas complementares e necessárias ao bom funcionamento do programa. De seguida, passar-se-á descrição funcional das mesmas.

5.1.1 Cliente MQTT (*ClientMain*)

O cliente MQTT é a classe principal do programa, responsável pela conectividade do mesmo. É nesta classe que se definem os parâmetros de conexão do programa ao *broker* instalado no servidor utilizado. Estes parâmetros são utilizados pelo programa para configurar a comunicação com o *broker* e assim garantir o bom funcionamento da mesma.

Os parâmetros de conexão definidos pelo programa são os seguintes:

- ***Broker***: endereço do *broker* ao qual o cliente se deve conectar;
- **Porto**: porto do *broker* ao qual o cliente se deve conectar. Importante, pois é diferente entre endereços não seguros TCP[4] e seguros SSL[5];
- **Tópico**: tópico(s) ao qual o programa deve subscrever no *broker*, de modo a garantir que todos os dados que chegam ao servidor são recebidos;
- ***QoS***: *QoS* com o qual o cliente se deve conectar ao *broker*;
- ***ClientID***: ID do cliente, para efeitos de identificação no *broker*. Não serve como *username* em caso de ligação segura ao *broker*.
- ***Username***: *username* (nome de utilizador) utilizado unicamente para efeitos de autenticação no servidor, em caso de ligação segura SSL;
- ***Password***: *password* (palavra-passe) utilizada unicamente para efeitos de autenticação no servidor, em caso de ligação segura SSL;
- ***CleanSession***: parâmetro muito importante. Caso seja definido como *true* (verdadeiro, ou seja, ativo num contexto prático), na próxima conexão ao *broker*, o cliente dá indicação para o *broker* apagar da sua memória todas as conexões passadas do cliente.

[4] TCP (do inglês *Transmission Control Protocol*) é um conjunto de regras (protocolo) utilizado com o *Internet Protocol* (IP) para enviar dados na forma de unidades de mensagem entre computadores através da Internet. Enquanto o IP trata do transporte dos dados propriamente dito, o TCP controla a localização das unidades individuais de dados em que a mensagem original é dividida (chamados *packets*) para otimizar o processo [53].
[5] SSL (do inglês *Secure Sockets Layer*) é um protocolo criptográfico de comunicação segura através da Internet muito comumente utilizado [38].

Caso contrário, a conexão retoma as ligações aos tópicos subscritos em ligações passadas.

O parâmetro *CleanSession* é extremamente importante pois permite salvaguardar uma possível desconexão inesperada do cliente. Por outras palavras, se o cliente se desconectar do *broker*, este parâmetro, quando definido como inativo (*false*) faz com que aquando da nova conexão ao *broker*, todas as subscrições sejam retomadas. Isto, aliado a um *QoS* superior a 0 (como explicado na secção 3.6) garante que todas as mensagens enviadas para o servidor enquanto o *broker* esteve desligado sejam recebidas.

Os parâmetros acima mencionados controlam o modo como o programa se conecta ao *broker*, instalado no servidor. A sua configuração é fundamental para garantir que o programa funciona como desejado.

Após definidos os parâmetros e iniciado o programa, o cliente é conectado e em caso afirmativo é apresentado a seguinte mensagem na consola:

```
ClientMain (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (6 de Set de 2013 14:01:43)
Connected to tcp://84.91[redacted]:1883 with client ID gcl-subscriber
Subscribing to topic "isaMQTTUser/test/isa/#" qos 1
Press <Enter> to exit
```

Figura 17: Ecrã de confirmação de conexão do cliente MQTT ao broker

Esta mensagem representa a confirmação de que a ligação foi estabelecida com sucesso ao servidor, além de apresentar informação acerca do *ClientID* utilizado e do *QoS* com que foi efetuada a ligação. Após estar conectado, toda a mensagem recebida é processada pelo método interno de tratamento de mensagens recebidas. Neste método, a mensagem é convertida numa *String* de texto e enviada para a classe (um módulo externo ao cliente MQTT, mas parte do mesmo programa) que descodifica a mensagem recebida em formato *JSON*.

5.1.2 Descodificador *JSON* (*ReadJSON*)

Esta classe, externa à classe principal Cliente MQTT, é responsável por descodificar as mensagens recebidas e extrair os dados que nelas vêm contidos. A classe é chamada pela classe principal Cliente MQTT sempre que uma mensagem é recebida para que o processo seja desenrolado, sendo fundamental para o objetivo do programa.

Para descodificar a mensagem, esta classe serve-se de uma livraria externa, a gson-2.2.2 [47], que contém as ferramentas necessárias a este processo - através desta ferramenta é possível ler os campos da mensagem, e assim extrair toda a informação que esta contém. Os dados obtidos da mensagem são guardados internamente em variáveis temporárias, para posteriormente serem enviados a outra classe, a de **escrita para ficheiro temporário**, que regista a informação num ficheiro adequado. As variáveis utilizadas são assim:

- IMEI;
- *aq_stamp* (*timestamp* da aquisição dos dados);
- *in_stamp* (*timestamp* do início da monitorização);
- *type* (tipo de medida efetuada);
- *aq_rate* (taxa de aquisição do aparelho que recolheu os dados em questão);
- *value* (valor da medida efetuada).

Definidas as variáveis, é chamada uma terceira classe denominada *FileWrite*, responsável pela escrita dos dados para ficheiros temporários. É enviada para esta classe toda a informação

contida nas variáveis supramencionadas, extraída da mensagem original, para que esta possa ser armazenada em ficheiros.

5.1.3 Escrita de Dados em Ficheiro Temporário (*FileWrite*)

A classe externa *FileWrite* recebe *Strings* com a informação extraída da mensagem original, já descodificada, e trata de a escrever para um ficheiro do tipo CSV[6], pequeno, de fácil leitura, escrita e organização. O ficheiro é temporário pois posteriormente sofre uma verificação de consistência dos dados, e só aí é escrita a versão final do mesmo.

Cada ficheiro temporário tem um cabeçalho identificativo da monitorização cujos dados armazena, seguido do total de dados da monitorização. Assim sendo, o processo de escrita do ficheiro temporário decorre em duas etapas (cujos procedimentos estão resumidos na Figura 18) descritas do seguinte modo:

1. **Verificação da existência de ficheiro**: o programa verifica se já existe algum ficheiro temporário para a monitorização em curso. Caso esta seja uma nova monitorização, o programa cria um novo ficheiro temporário. O nome do ficheiro tem a seguinte composição:
    - IMEI do aparelho que faz a coleta de dados (por exemplo um *tablet*), seguido da data de início da monitorização (em milissegundos), unidos por um *underscore* (_);
    - Sufixo *temp* (também precedido por um *underscore*). Por exemplo, 343b289f7fc2adb2_1377037166769_temp.csv.

    Uma vez garantido que o ficheiro temporário existe, o programa escreve um cabeçalho no ficheiro, com o IMEI numa linha e a **data de início de monitorização** na seguinte. Por fim é adicionada uma linha de legenda com o título de cada campo de dados a ser posteriormente armazenados em colunas (TYPE na coluna 1, AQUISITION TIME GAP na coluna 2, AQUISITION RATE na coluna 3 e VALUE na coluna final).
2. **Adição de dados**: depois de confirmada a existência do ficheiro temporário, o que o programa faz é adicionar os dados ao mesmo. Este processo tem como base a mensagem recebida, cujos elementos foram previamente descodificados, e passa por escrever esses elementos em colunas do ficheiro CSV:
    - TYPE (Tipo): Tipo de medida. Uma vez que o programa é capaz de ler diferentes tipos de dados, e que na mesma monitorização possam ser medidos diferentes sinais biomédicos, torna-se fundamental que para cada valor de dados adicionado, o tipo de medida que que foi efetuada seja identificada. Deste modo, a classificação de tipos é a seguinte:
        1. Oximetria de Pulso: Saturação de Oxigénio no sangue arterial, obtida através do Oxímetro de Pulso Nonin 4100. A unidade é $\%SpO_2$.
        2. Frequência Cardíaca (Oxímetro): Frequência cardíaca obtida através do Oxímetro de Pulso Nonin 4100. As unidades são Batimentos Por Minuto (BPM).
        3. Temperatura: Temperatura adquirida através de um sensor incluído no produto Plux BioSignals utilizado. As unidades são graus Celsius (º).
        4. ECG: Valores de Eletrocardiograma obtidos através de sensor componente do produto Plux BioSignals utilizado. As unidades são Volts (V).
        5. Frequência Cardíaca: Frequência cardíaca obtida através do produto Plux BioSignals utilizado. As unidades são Batimentos Por Minuto (BPM).

[6] CSV (em inglês, *Comma Separated Values*) é um tipo de ficheiro que permite armazenar dados em tabelas, em que os valores de cada linha são separados por uma vírgula ou outro caracter definido.

6. Taxa Respiratória: Frequência respiratória adquirida através de sensor incluído no produto Plux BioSignals. As unidades são Volts (V).

- AQUISITION TIME GAP (Intervalo entre data de aquisição e data inicial de monitorização, em milissegundos): Intervalo entre a data inicial de monitorização e a data do momento de aquisição dos dados pelo dispositivo que coleta os dados dos sensores. Este é o valor teórico esperado do intervalo, sendo por isso definido pelo aparelho que enviou os dados, em função da taxa de aquisição do Tipo de Medida (utiliza-se o inverso da taxa de aquisição, que é adicionado incrementalmente, em milissegundos).
- AQUISITON RATE (Taxa de Aquisição – em Hertz): Taxa a que os dados são adquiridos pelo sensor.
- VALUE (Valor): Valor efetivo da medida em causa.

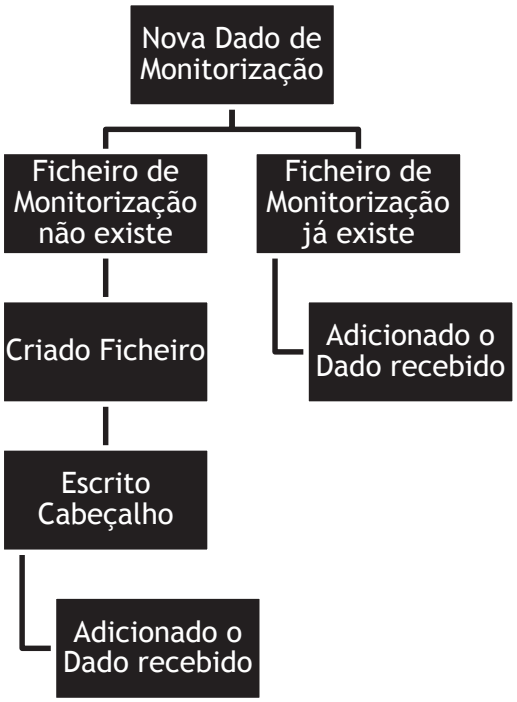


Figura 18: Resumo do processo de escrita de ficheiro

De cada vez que uma mensagem com dados chega, é adicionada uma nova linha ao ficheiro temporário CSV com os 4 primeiros campos preenchidos com a informação correta (TYPE, AQUISITION TIME GAP, AQUISITION RATE e VALUE).

5.1.4 Organização Final de Dados no Ficheiro (*FileOrganize*)

A classe *FileOrganize* é chamada de 10 em 10 segundos pela classe principal *ClientMain*, para verificar se os ficheiros acabados de receber estão devidamente ordenados de acordo com o momento de aquisição, e de escrever para ficheiro final esses mesmos dados.

A organização temporal de dados é fundamental para garantir a coerência das monitorizações. O funcionamento do sistema de aquisição e envio de dados para o programa está resumido na Figura 19, que permite uma melhor compreensão da importância desta questão.

Este sistema apresenta uma característica potencialmente limitante, consequência do uso do *tablet* como dispositivo de coleta de dados, à qual foi dada extrema atenção. Se o processador do *tablet* ou o próprio aparelho sensor (por qualquer motivo) ficar lento, verificar-se-á uma diferença temporal (um atraso) entre o momento real de aquisição e o momento em que o

tablet recebe e processa a informação. Se o *tablet* atribuísse como momento de aquisição a data em que a informação lhe chega, poderia produzir-se um erro consequente do tempo que demorou ao sensor a enviar a informação, ou do tempo decorrido entre a receção da medida pelo *tablet* e o processar desta informação. Apesar do valor do erro poder não ser significativo em alguns casos, os dados seriam afetados, comprometendo a fidelidade da análise e visualização das monitorizações. Como se trata de um assunto de extrema delicadeza, a saúde de um ser humano, foi necessário desenvolver uma solução para esta limitação.

Figura 19: Processo de aquisição e envio de dados biomédicos para servidor

A solução encontrada tem como base o conhecimento da **taxa de aquisição** de cada sensor. Sabendo a quantidade de dados que o sensor adquire num intervalo de tempo fixo, é possível calcular o tempo que decorre entre cada aquisição. Quando os dados coletados são processados pelo *tablet*, este define a data de aquisição através do inverso da taxa de aquisição da medida em causa: acrescenta sempre ao valor anterior o inverso da taxa de aquisição. Deste modo, garante-se que os tempos correspondem aos tempos exatos de aquisição pelo sensor.

Por exemplo, dados de Oximetria de Pulso obtidos através do Oxímetro de Pulso *Nonin 4100* são adquiridos com uma frequência de 1Hz (um valor a cada segundo - mil milissegundos). O primeiro valor teria tempo de aquisição 0, o segundo 1000, o terceiro 2000, e assim sucessivamente. Para obter o valor real da data de cada medida, basta somar este intervalo de tempo à data de início da monitorização.

Chamada de 10 em 10 segundos, o que a classe *FileOrganize* faz é verificar se existem ficheiros temporários já finalizados (ou seja, cuja monitorização correspondente já tenha terminado) e ler todos os dados no ficheiro temporário e ordenando-os por data de aquisição crescente. O ficheiro definitivo é igual ao temporário mas de organização garantida. O processo de organização passa-se do seguinte modo:

1. É lido o ficheiro temporário, e extraída toda a informação do mesmo;
2. Ordena-se cronologicamente a informação extraída do ficheiro temporário;
3. É criado o novo ficheiro, cuja estrutura é em tudo igual ao original;
4. São escritos para o ficheiro final todos os dados, já devidamente ordenados. Quando finalizada a escrita o ficheiro temporário é eliminado, sobrando apenas o ficheiro final que possui o nome do temporário original, sem o sufixo *_temp*.

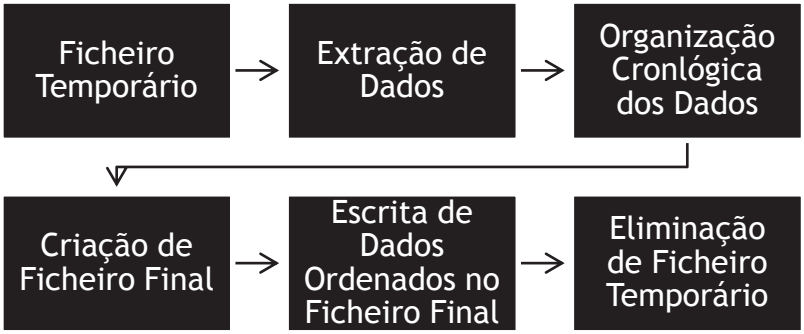


Figura 20: Etapas para criação do ficheiro final

5.2 Ferramenta de Visualização de Dados

A **ferramenta de visualização de dados** permite, como o nome indica, visualizar os dados armazenados previamente, e assim aprofundar a análise dos mesmos. Funcional e estruturalmente divide-se em duas partes, em que a primeira - um programa escrito em Java - serve de base à apresentação de dados na segunda – a interface gráfica da ferramenta, escrita em HTML e com um JavaScript responsável pela representação gráfica dos dados. Para desenvolver, testar e utilizar a **ferramenta** recorreu-se ao *software Apache Tomcat*, instalado com as definições originais e utilizado nas suas funcionalidades elementares.

O *Apache Tomcat* serve principalmente como servidor de aplicações JEE (*Java Enterprise Edition*) - que é o caso da **ferramenta de visualização -** tendo também capacidade de funcionar como servidor Web. O Eclipse, IDE utilizado para desenvolvimento da ferramenta, permite incorporar o *Tomcat* nos seus recursos e assim efetuar a sua gestão automaticamente - ligar, desligar, e atualizar quando necessário. Para informação sobre este *software* consultar [48].

A organização funcional da ferramenta pode ser resumida de acordo com a Figura 21.

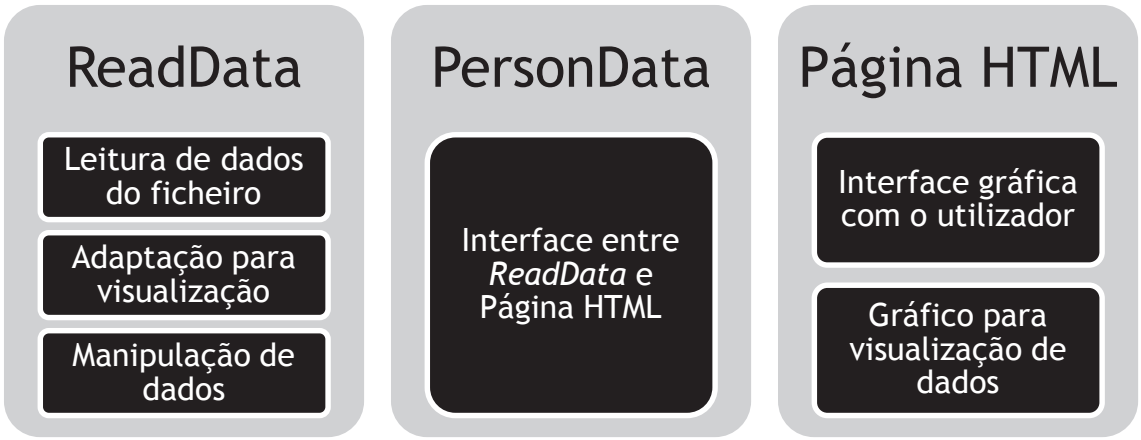


Figura 21: Estrutura global da ferramenta de visualização de dados

A classe *ReadData* desempenha a função de pré-tratamento dos dados. Todas as tarefas de aquisição, manipulação e preparação dos dados são efetuadas por esta classe, que assume assim o papel de força trabalhadora do código.

A classe *PersonData* funciona como interface entre a classe *ReadData* e a Página HTML. De cada vez que o utilizador da Página HTML requer alguma função a desempenhar pela classe *ReadData*, a Página HTML interage com a classe *PersonData*, que por sua vez efetua a comunicação com a classe *ReadData*. Além disso, a classe *PersonData* armazena informação fundamental ao funcionamento da Página HTML, fornecida pela classe *ReadData* ou introduzida pelo utilizador na Página HTML.

A Página HTML é a interface gráfica com o utilizador, i.e. com o exterior do meio de programação. É nesta página que são apresentadas ao utilizador as opções que este pode escolher (por exemplo o ficheiro que deseja abrir), bem como o gráfico resultante da análise dos dados importados do ficheiro de monitorização.

5.2.1 Leitura de ficheiro de Dados - *ReadData*

Para compreender bem o funcionamento da ferramenta de visualização de dados, é necessário o conhecimento do funcionamento das suas diversas partes. A primeira é a que corresponde à importação dos dados contidos no ficheiro que se pretende visualizar - a classe *ReadData*.

Inicialmente é importante compreender a organização dos dados aquando da sua importação. Após devidamente importados, os dados devem estar organizados de modo a garantir que o programa os consegue utilizar. De seguida passar-se-á à compreensão da estrutura intermédia, a classe *PersonData* que disponibiliza toda a informação para a ponta final da ferramenta, a Página HTML, responsável pela apresentação de informação ao utilizador e por receber instruções do mesmo.

5.2.1.1 Estrutura Interna dos Dados na classe ReadData

Para melhor organização dos dados importados, e consequente melhor funcionamento do programa, os dados são estruturados internamente pela classe *ReadData* em três classes diferentes:

- *Measurement*: corresponde a uma **monitorização** completa, i.e. a toda a informação que torna única cada monitorização, incluindo os valores obtidos durante cada uma;
- *Measurement Value:* que corresponde a cada **medida** existente numa dada **monitorização,** e a toda a informação relativa à mesma (**tipo de medida**, **data de aquisição**, **taxa de aquisição,** e **valor**). Por exemplo, um valor de Oximetria de Pulso (%98 SpO_2) e a informação relativa a esta;
- *Measurement Type*: corresponde a cada **tipo de medida**, i.e. ao tipo de exame que a que corresponde a uma dada medida. Cada instância desta classe contém toda a informação relativa ao tipo de medida que identifica (além do nome, possui também informação relativa às unidades respetivas). Por exemplo uma medida de Eletrocardiograma terá como nome do tipo de medida "Eletrocardiograma" e unidades "Volts".

Por outras palavras, considera-se **medida** (*Measurement Value*) cada valor individual de um dado **tipo de medida**. Por exemplo, se existirem 30 valores de Oximetria de Pulso numa dada **monitorização**, há 30 **medidas** (portanto 30 instâncias de *Measurement Value*) do **tipo de medida** (*Measurement Type*) Oximetria de Pulso nessa **monitorização**. Daqui em diante, apenas será mencionada a nomenclatura correspondente em português.

A estrutura interna das **classes** (e consequentemente dos dados) segue a seguinte organização:

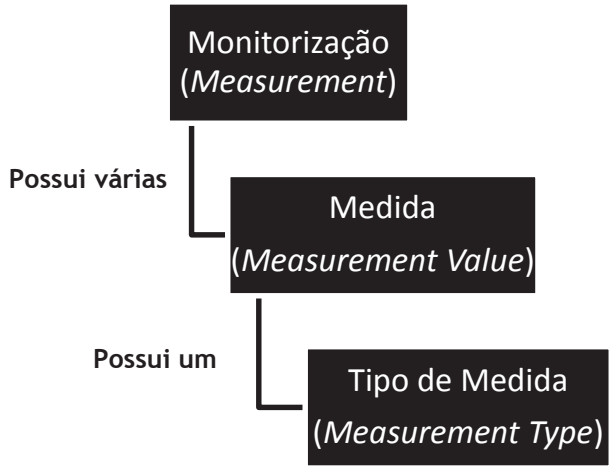


Figura 22: Estrutura interna de classes para importação de dados

Cada **monitorização** possui várias **medidas**, que por sua vez possuem apenas um **tipo de medida**. Mas para ficarem bem definidas, estas três classes possuem atributos internos que devem ser preenchidos com informação. Na Tabela 1 apresentam-se os diferentes atributos, ou seja campos de informação, de cada uma destas categorias.

Tabela 1: Classes principais utilizadas na importação de dados

Categoria	Atributos
Monitorização	ID (um número inteiro de identificação)
	Data de Início (em milissegundos)
Medida	ID (um número inteiro de identificação)
	Data de Aquisição (em milissegundos)
	Valor (em texto)
Tipo de Medida	ID (um número inteiro de identificação)
	Nome (do tipo – exemplo: ECG)
	Código (uma inicial para identificação)
	Unidade (de medida – exemplo: Volts)
	Taxa de Aquisição (exemplo: 100Hz)

As três classes supramencionadas são utilizadas na fase de importação de dados para o programa.

No entanto, são necessárias classes adicionais de modo a preparar estes dados para visualização no JavaScript (a **ferramenta de gráficos**). Estas serão utilizadas aquando do armazenamento dos dados em variáveis co o intuito de facilitar o acesso, manipulação e posterior visualização dos respetivos dados. Assim, as classes adicionais são a *DataValue*, *DataValues* e *DataSeries*, e têm o seguinte objetivo:

- ***DataValue*:** classe que agrega os dados necessários para visualização de cada **medida**. Como atributo tem apenas uma matriz (um *Array*) de objetos, que irá preencher a primeira posição com o valor da **data de aquisição** em milissegundos, e a segunda posição com o **valor da medida** (na forma de um número do tipo *Double*). Cada *DataValue* corresponderá assim a um ponto num gráfico de medidas;
- ***DataValues*:** lista de objetos do tipo *DataValue*. É assim uma lista das **medidas** a serem apresentados no gráfico final da ferramenta;
- ***DataSeries*:** classe que visa ser o contentor final de um determinado conjunto de dados, ou seja de **medidas**. Cada classe *DataSeries* contém valores de medidas correspondentes a apenas um **tipo de medida** (exemplo: Eletrocardiograma). Possui três campos que são preenchidos de acordo com os dados lidos:
    - **Nome:** nome da série de dados, correspondente ao tipo de dados que contém (exemplo: Eletrocardiograma);
    - **Unidade:** unidade de medida dos valores contidos na série de dados (exemplo: Volts);
    - ***DataValues*:** classe com os valores obtidos da medida em causa, tal como explicado acima.

Aquando do envio de dados para a **ferramenta de gráficos**, estes são enviados sob a forma de uma *String JSON*, que possui uma lista das *DataSeries* a serem visualizadas. Cada *DataSeries* será uma série de dados diferente no gráfico final, algo melhor compreendido aquando da apresentação deste componente.

O último elemento a realçar é a classe ***mvData***, uma classe interna da classe *ReadData*. A classe *mvData* funciona como uma ferramenta interna que facilita a manipulação e acesso aos dados obtidos aquando da interação da *ReadData* com a classe *PersonData*. Assim, a *mvData* tem as seguintes variáveis internas:

- Uma **lista de medidas**, definida aquando da leitura dos ficheiros;
- Uma **lista de *DataSeries***, criada a partir da lista de medidas;

- O valor da **taxa de aquisição** mais elevada de todas as identificadas no ficheiro.

Ao ser utilizada internamente pela classe *ReadData* para armazenar dados, a classe *mvData* permite uma manipulação de dados mais fácil, bem como um melhor desempenho da classe *ReadData* – nomeadamente quando o utilizador deseja subdividir o total de dados em intervalos, facilitando a representação dos mesmos na ferramenta de gráficos.

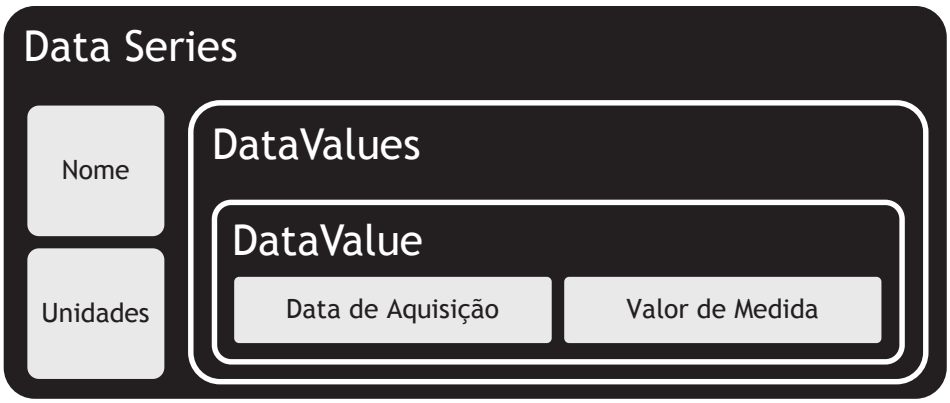


Figura 23: Hierarquia de classes principais utilizadas na importação de dados

5.2.1.2 Criação de Lista de Medidas por Leitura do Ficheiro

O primeiro passo para visualizar os dados é importar os mesmos para o programa. Esta etapa do programa serve-se de várias classes que coletivamente desempenham as tarefas que permitem ler os dados e disponibilizá-los para posterior utilização da ferramenta.

Para proceder à importação de dados contidos em ficheiros, o programa serve-se de uma livraria (openCSV-2.3 [49]) que proporciona, entre outras funções, a leitura de todas as linhas do ficheiro do tipo CSV, uma a uma. Ao ler o cabeçalho do ficheiro, o programa extrai as seguintes informações:

- IMEI;
- Data de início da aquisição (em milissegundos) (campo *INITIAL TIME* no ficheiro).

Com esta informação o programa pode criar uma nova **monitorização**, dado que já consegue definir os seus atributos. Criada e devidamente definida a monitorização em que se vão inserir os dados importados do ficheiro, o programa continua a leitura das linhas do ficheiro.

Na terceira linha do ficheiro estão os títulos dos campos preenchidos abaixo com dados obtidos dos sensores (*TYPE*, *AQUISITION TIME GAP*, *AQUISITION RATE*, *VALUE*, *REAL TIME GAP*), uma legenda destinada a facilitar a leitura do ficheiro caso este seja aberto com um editor de texto ou folha de cálculo. A partir da quarta linha está a informação correspondente a todas as medidas efetuadas durante a monitorização, sendo que cada linha tem informação relativa a uma **medida**. De cada vez que uma nova linha do ficheiro CSV é lida são extraídos os quatro atributos presentes na linha, iniciando-se o processo de criação de uma nova **medida** no programa. No final deste processo esta é adicionada a uma lista.

O processo de criação de uma medida decorre do seguinte modo:

1. É criada a nova **medida** (*Measurement Value*), com os campos vazios;
2. Define-se o valor da **data de aquisição**, extraída do ficheiro (soma: INITIAL TIME + AQUISITION TIME GAP);
3. Define-se o **valor da medida** na forma de texto (*String*), extraído do ficheiro;

4. Define-se o **ID da medida**, com base no número de medidas já adicionadas. Ou seja, a primeira medida tem o ID = 1, a segunda tem ID = 2, e assim sucessivamente;
5. Define-se a **monitorização** em que a medida se insere (a mesma que foi criada aquando da identificação do cabeçalho);
6. Define-se o **tipo de medida** em causa. É extraído o identificador do **tipo de medida** a partir do ficheiro, e depois este é utilizado para associar o tipo de medida correto à medida em causa.
7. Percorridos todos os passos anteriores, a **medida**, com todos os atributos preenchidos, é adicionada à lista de Medidas desta Monitorização;

Quando o programa identifica que todas linhas do ficheiro já foram lidas, termina a sua leitura e adiciona a **lista de medidas** recém-criada a uma nova instância da classe *mvData* (sempre que é efetuada a leitura de um novo ficheiro, é criada uma nova instância desta classe). Aquando da criação da lista de valores, esta é adicionada à instância da classe criada como uma variável interna – atributo - da classe. De seguida é definido como outro atributo da classe *mvData* o valor máximo de **taxa de aquisição** de dados lido do ficheiro. Fica apenas a faltar definir a lista de *DataSeries*, mas tal é efetuado no passo seguinte da importação de dados do ficheiro.

A importância de utilizar toda esta estrutura prende-se não só com o facto de esta garantir um bom funcionamento do programa, mas também com o facilitar da incorporação do trabalho com a tecnologia já existente do produto OneCare.

5.2.1.3 Criação de Lista de DataSeries a partir de Lista de Medidas

Finalizado o processo de criação da **lista de medidas**, o programa chama um novo método para transformar esta **lista de medidas** global (ou seja, com todos os dados lidos no ficheiro) num objeto mais organizado e preparado a ser lido pelo JavaScript da ferramenta de gráficos.

Recebendo a **lista de medidas** criada no passo anterior como um argumento, este método vai converter esta mesma numa nova lista mas desta vez, uma lista de *DataSeries*. O método começa por criar duas variáveis:

- Uma **lista** de *DataSeries*, que será o elemento final a ser produzido por este método;
- Um **mapa** de *DataSeries*, que facilita de cada vez que se pretende aceder a uma *DataSeries* presente (ou que se deseje que exista) na lista.

De seguida, são lidas as **medidas** da lista de medidas recebida. Para cada **medida**, o método efetua o seguinte procedimento:

1. Obtém o **tipo de medida** associado à medida em leitura;
2. Verifica (no mapa criado inicialmente) se já existe alguma *DataSeries* para o **tipo de medida** obtido no ponto 1. Caso afirmativo, procede para o passo 3. Em caso negativo, segue as seguintes etapas:
    a. É criada uma nova *DataSeries* com o **nome** do **tipo de medida** obtido no ponto 1 e tendo como **unidades** as unidades respetivas do tipo de medida em questão.
    b. É criada uma instância de *DataValues* para esta nova *DataSeries*, apta a ser preenchida por objetos do tipo *DataValue*, correspondentes a cada medida deste tipo.
    c. A recém-criada *DataSeries* é adicionada ao mapa de *DataSeries*, criado originalmente pelo método;
    d. A mesma *DataSeries* é adicionada à lista de *DataSeries* criada previamente pelo método;

3. É acedida a instância de *DataValues* à qual se pretende adicionar o *DataValue* correspondente à **medida** em leitura (*DataValue* esse ainda por definir). A partir deste momento, o programa sabe onde deve adicionar o *DataValue* correspondente à medida lida;
4. É criado o *DataValue* a ser adicionado à *DataValues* acedida em 3 (e consequentemente à *DataSeries* desejada). O *DataValue* criado é uma matriz de objetos, que recebe como primeiro objeto a **data de aquisição da medida**, e como segundo objeto o **valor** (em formato *Double*) da medida.
5. O *DataValue* é adicionado à *DataValues* acedida no ponto 4, e consequentemente à *DataSeries* que a possui (a detetada ou criada no ponto 2).

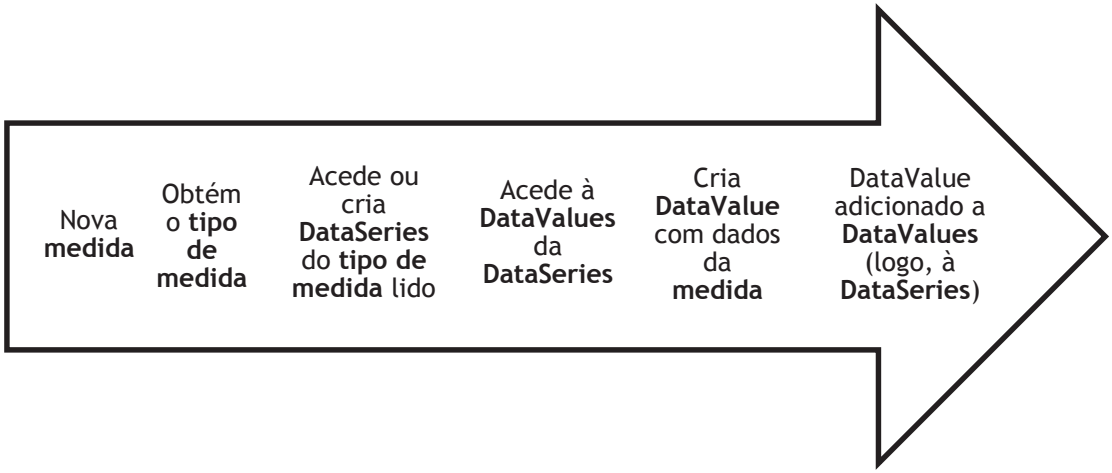


Figura 24: Processo de criação de DataSeries

No final, aquilo que se obtém é uma lista com tantas *DataSeries* quantos tipos de medida encontrados na lista de medidas obtida através da importação de dados do ficheiro. Cada *DataSeries* contém todos os valores das medidas, de um determinado tipo de medida, feitas durante a monitorização cujo ficheiro foi lido. Segue-se um exemplo da estrutura final de uma lista de *DataSeries* criada após os procedimentos supramencionados. Para esta lista, considera-se que durante a monitorização foram adquiridos dados de Oximetria de Pulso, Eletrocardiograma e Temperatura durante 60 segundos. A estrutura final seria do seguinte tipo:

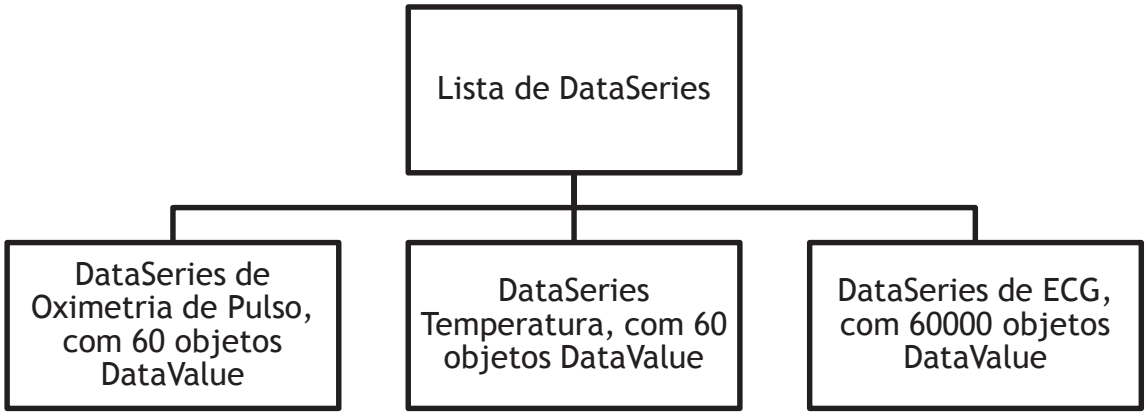


Figura 25: Exemplo estrutural de lista de DataSeries

A lista final de *DataSeries* possui todas as *DataSeries* que foram criadas pelo método, e é adicionada à instância de *mvData* criada anteriormente. Esta instância fica assim com todos os atributos definidos (a **lista de medidas**, a **lista de *DataSeries*** e a **taxa de aquisição máxima**). É através desta lista de DataSeries que serão gerados os valores para visualização gráfica.

5.2.1.4 Adaptação a Objeto JSON para Visualização Gráfica

A próxima etapa passa por adaptar os dados, para que sejam corretamente lidos pelo JavaScript da **ferramenta de gráficos**. Para tal existe um método que recebe (como argumento) a lista de *DataSeries* criada no passo anterior e a converte numa *String* de texto, seguindo a codificação do tipo *JSON* (*JavaScript Object Notation*).

Para converter a Lista de Data series na *String JSON* o método utiliza uma livraria, a gson-2.2.2 [47]. Esta pequena ferramenta contém todos os recursos necessários para fazer a conversão de uma maneira simples e eficaz, sem necessidade de tornar o programa mais complexo. O processo de conversão dos dados está esquematizado na Figura 26.

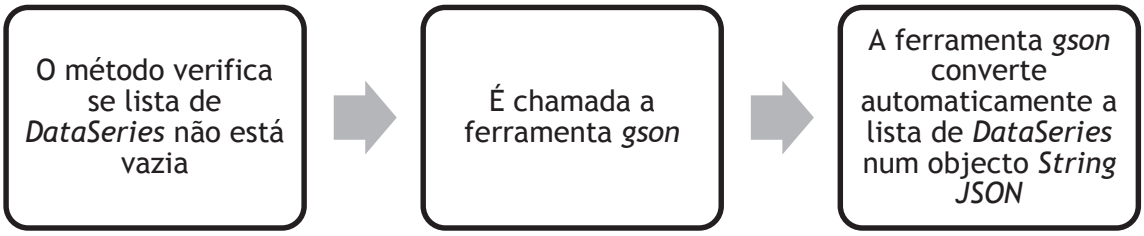


Figura 26: Resumo do processo de conversão de lista de DataSeries em String JSON

No final é obtida uma *String* definida segundo os parâmetros de **objeto *JSON***, pelo que uma ferramenta que interprete (leia e escreva) *Strings JSON* já consegue ler, descodificar e extrair a informação do objeto criado. A *String JSON* será enviada para o JavaScript da ferramenta de gráficos, que com desta extrai a informação e assim permite a visualização gráfica dos mesmos.

5.2.2 Manipulação de Dados - *ReadData*

Apesar de os dados anteriores permitirem visualizar os dados em bruto, é necessário tomar outras considerações em conta para garantir um bom funcionamento do programa e evitar erros durante a visualização de dados. Nesse sentido, os seguintes procedimentos foram implementados:

- Obtenção de **lista de ficheiros e pastas** disponíveis, para que o utilizador da ferramenta apenas tenha de escolher os dados desejados, em vez de os ter de inserir manualmente.
- **Divisão de dados por intervalos**, para garantir que não há sobrecarga da **ferramenta de gráficos** com excesso de dados. Para conseguir este efeito, é necessário um passo intermédio:
    - Obtenção do intervalo que se pretende visualizar. Tal é feito recebendo a página e **taxa de aquisição máxima** do conjunto de dados a visualizar;
    - Obtenção do total de intervalos, para que a Página HTML o possa comunicar ao utilizador;

De seguida são explicados os métodos referidos.

5.2.2.1 Lista de Ficheiros e Pastas

Este passo, embora não fulcral para o funcionamento do programa, é muito útil pois evita que o nome do ficheiro a ser lido seja mal inserido e provoque erros no programa. A interface gráfica serve-se desta pequena ferramenta para fornecer ao utilizador uma lista de pastas e ficheiros (existentes nessas pastas). Isto permite que a organização dos ficheiros seja mais eficaz e assim o acesso aos dados seja mais sistemático e claro.

O método (inserido na classe *ReadData*) serve-se de código Java para identificar as pastas e ficheiros contidos na pasta raiz onde o primeiro programa guarda os dados. Consoante a instrução que é dada a este método, este ou devolve uma **lista de pastas** existentes na pasta raiz pré-definida pelo programa, ou devolve uma **lista de ficheiros** existentes na pasta selecionada (escolhida da lista já criada pelo próprio método, ou a pasta raiz - se pretendido ou caso nenhuma pasta seja selecionada).

Ambas as listas são devolvidas independentemente, pelo que é necessário utilizar o método separadamente para obter a lista de pastas ou a lista de ficheiros, como desejado. Isto acontece para permitir ao utilizador escolher a pasta onde deseja pesquisar ficheiros.

5.2.2.2 Divisão de Dados por Intervalos

Um pormenor muito importante do programa é a capacidade de este identificar automaticamente quando é necessário dividir e mostrar os dados por intervalos, para não sobrecarregar a **ferramenta de gráficos** (o JavaScript, descrito mais adiante na secção 5.2.3.2). Em sistemas de visualização gráfica como este, que lidam com muita informação, a quantidade significativa de dados pode contribuir para a sobrecarga da ferramenta gráfica, podendo esta deixar de funcionar. Para evitar a saturação do JavaScript, existem dois métodos muito importantes:

- O método que efetivamente divide os dados em intervalos de duração bem definida;
- O método que calcula o número de intervalos nos quais os dados serão divididos, tendo em conta a **taxa de aquisição** máxima dos mesmos.

A importância de utilizar a **taxa de aquisição máxima** de dados prende-se com o seguinte facto: caso a lista de dados (mais tecnicamente, a lista de *DataSeries*) possua vários **tipos de medida**, aquele que possuir uma maior taxa de aquisição é o que mais vai contribuir para preencher o mesmo intervalo de tempo no gráfico. Logo deve ser o elemento limitante no que toca ao total de dados que o gráfico pode expor, uma vez que é o elevado número de dados que pode levar ao mau funcionamento da **ferramenta de gráficos**.

É assim relevante destacar primeiro este método que calcula o número de intervalos em que o total de dados pode ser dividido, bem como a forma como tal é conseguido.

5.2.2.2.1 Cálculo do Número Total de Intervalos de Dados

O número total de intervalos (ou páginas) no qual os dados lidos podem ser divididos é obtido através da **lista de *DataSeries*** e da **taxa de aquisição máxima**, ambas armazenadas na classe *mvData*. Cada intervalo terá a duração correspondente a quinhentas medições, no máximo. Limitar-se-á assim o máximo de dados fornecidos ao JavaScript da ferramenta de gráficos, evitando que este incorra em mau funcionamento.

O método percorre todos os dados da lista de *DataSeries* e extrai do mesmo a data de início de monitorização e a data de final de monitorização. Este é o total temporal que será dividido em intervalos em função do máximo de dados existentes nesse mesmo intervalo, valor este que corresponde ao total de dados da *DataSeries* de maior taxa de aquisição.

Conhecido o total temporal, o próximo passo é a sua divisão. Nesta fase, será utilizada a **taxa de aquisição máxima**, definida aquando da leitura do ficheiro. As **datas** (e consequentemente o **intervalo** entre ambas) estão em milissegundos, e a taxa de aquisição está em Hertz (ou seja, s^{-1}), pelo que é necessária uma simples conversão unitária.

A equação que define o número de páginas total é a seguinte:

$$\frac{Data\ Final\ de\ Aquisição - Data\ Inicial\ de\ Aquisição\ (ms)}{5000000 / Taxa\ Máxima\ de\ Aquisição\ (Hz)}$$

$$\Rightarrow \frac{Intervalo\ (ms) \times Taxa\ Máxima\ de\ Aquisição\ (s^{-1})}{5000000}$$

O valor 5000000 corresponde a 500 dados por página. É um valor obtido experimentalmente para garantir o bom funcionamento da **ferramenta de gráficos**. Assim, o JavaScript fica funcionalmente otimizado, permitindo uma visualização clara dos dados e um maior detalhe na análise dos mesmos.

Como é possível verificar pela análise matemática da equação, as unidades temporais anulam-se, pelo que o valor final (sem unidades) é o número de páginas total. No entanto, ainda falta efetuar uma operação a este valor para obter o valor final e definitivo do total de intervalos: o valor deve ser arredondado, pois é extremamente improvável que a equação determine um valor inteiro. Acrescenta-se ainda um detalhe importante: o valor inteiro obtido deve ser resultado de um arredondamento por excesso e nunca por defeito, pois tal iria resultar numa exclusão de dados. Consideradas estas condições, o programa arredonda o valor obtido sempre para o valor inteiro imediatamente superior.

Deste modo garante-se que o número total de Intervalos não exclui dados e permite a visualização da totalidade dos dados divididos por intervalos. Este método é utilizado pela interface gráfica para o utilizador saber o total de páginas que pode navegar.

5.2.2.2.2 Criação dos Intervalos de Dados

Um dos processos mais importantes da manipulação de dados é a divisão dos mesmos em intervalos. Todos os intervalos são de igual tamanho (a nível de dados expostos), exceto o último intervalo, que expõe os dados que sobram da divisão, podendo assim ter um tamanho variável. Assim, o tamanho máximo de um intervalo é de 500 dados, sendo que o último intervalo pode possuir menos dados (exemplo: um total de 1320 dados divide-se em dois intervalos de 500 dados e um intervalo final de 320 dados).

É importante relembrar que os dados a ser divididos correspondem ao total de dados existentes na **lista de *DataSeries*** que o método recebe, criada aquando da leitura do ficheiro. Se existir mais do que um **tipo de medida** (ou seja, mais do que uma *DataSeries*), podem existir diferentes **taxas de aquisição**, pelo que este método não pode simplesmente fazer uma contagem de dados para definir os intervalos. Se assim fosse iria resultar num intervalo temporal diferente para *DataSeries* com diferentes **taxas de aquisição**, podendo levar a um mau funcionamento da **ferramenta de gráficos**. Para representar vários tipos de medidas num só gráfico verifica-se necessário que o intervalo de tempo de visualização (em que todas as *DataSeries* da lista são divididas) seja igual para todas. A quantidade de dados será assim o elemento que pode variar entre *DataSeries* da mesma lista, no mesmo intervalo temporal. Como referido aquando do cálculo do total de intervalos de divisão (secção 5.2.2.2.1), a *DataSeries* com valor mais elevado de **taxa de aquisição** irá definir o intervalo de tempo total do intervalo. A

Tabela 2 possui exemplos para melhor compreensão desta questão.

Compreendida a explicação por detrás da obtenção do tamanho dos intervalos, passa-se apresentação da divisão propriamente dita. O pressuposto inicial que suporta esta divisão é simples: os intervalos têm um tempo de início e final bem definido. O programa limita-se a verificar quais os valores de cada *DataSeries* que estão compreendidos neste intervalo.

Para cada *DataSeries* original é criada uma nova *DataSeries* com apenas os valores que se encontram no intervalo desejado, e no final todas as *DataSeries* criadas são adicionadas a uma nova lista de *DataSeries*. De cada vez que se pretende visualizar um novo intervalo, este método é chamado, efetuando este procedimento e daqui resultando uma lista com *DataSeries* que contém os dados correspondentes ao intervalo temporal que se deseja visualizar.

Tabela 2: Exemplos de definição de intervalo de dados para visualização

Lista	*DataSeries*	Taxa de Aquisição	Tamanho do Intervalo [a]	Dados por Intervalo Normal
Lista com 2 *DataSeries*	ECG	100Hz	5 Segundos	500
	Oximetria de Pulso (Plux)	1Hz		5
Lista com 1 *DataSeries*	Oximetria de Pulso (Oxímetro)	1Hz	500 Segundos	500
Lista com 3 *DataSeries*	ECG	100Hz	5 Segundos	500
	Taxa Respiratória	10Hz		50
	Oximetria de Pulso (Plux)	1Hz		5

[a] Definido pela *DataSeries* de maior valor.

No final, resultam apenas os dados correspondentes às **medidas** efetuadas no intervalo de tempo cuja visualização é pretendida. A lista de *DataSeries* original é preservada, para que os dados se mantenham disponíveis para futura manipulação: por exemplo numa nova intenção de visualizar um intervalo diferente, é novamente acedida à lista original, mudando apenas o intervalo temporal de dados que se pretende visualizar.

Compreendidos os processos que levam à importação dos dados, segue-se a apresentação das ferramentas responsáveis pela interface visual com o utilizador e de como estas interagem com a classe *ReadData*.

5.2.3 Visualização Gráfica de Dados

Até agora, apenas foi detalhada a importação dos dados a partir de ficheiros, faltando compreender como é que esta informação é passada à **ferramenta de gráficos**. Para este efeito existe a classe *PersonData*, que comunica com a classe *ReadData* e ao mesmo tempo com a interface gráfica apresentada ao utilizador.

5.2.3.1 Interface com a Parte Gráfica - PersonData

A classe *PersonData* funciona como intermediária entre a classe *ReadData* - que opera as tarefas fundamentais para disponibilização da informação contida nos dados - e a interface com o utilizador. Esta classe tem duas funcionalidades principais:

- Armazenamento e disponibilização de informação;
- Execução de métodos: maioritariamente *pedidos* à classe *ReadData*, para que esta execute as suas capacidade e devolva informações extraídas dos ficheiros.

A funcionalidade de armazenamento é fundamental, pois permite o acesso direto aos dados biomédicos presentes nos ficheiros.

A informação armazenada é a seguinte:

- **Total de páginas** que a ferramenta de gráficos irá expor. Cada uma tem no máximo quinhentas medidas;
- Página a ser visualizada no momento, pelo utilizador;
- ***String JSON*** com dados prontos para a ferramenta de gráficos: a classe *PersonData* requere à classe *ReadData* que esta leia os dados do ficheiro e os devolva. Depois da classe *ReadData* ler os dados, estes são enviados para a classe *PersonData* já prontos a serem disponibilizados para a ferramenta de gráficos. Na classe *PersonData*, os dados estão assim disponíveis para que a *Página HTML* lhes aceda e os *forneça* ao JavaScript (a ferramenta de gráficos).
- **Lista de *DataSeries*** existentes no ficheiro em leitura, obtida através da classe *ReadData*;
- **Mapa de pastas** existentes da pasta raiz, onde se encontram os ficheiros para leitura. Obtido a partir da classe *ReadData*;
- **Pasta selecionada** pelo utilizador. Este dado provém da **interface gráfica** com o utilizador;
- **Mapa de ficheiros** existentes na **pasta selecionada** pelo utilizador. Obtido a partir da classe *ReadData*;
- **Ficheiro selecionado** pelo utilizador. Este dado provém da **interface gráfica** com o utilizador;
- ***Caminho*** para o **ficheiro selecionado**. Esta informação é enviada à classe *ReadData* e identifica a localização exata do ficheiro cuja leitura é pretendida.

Os métodos presentes são de extrema importância pois permitem a obtenção dos dados armazenados na classe *PersonData* (através de *getters*, métodos específicos que devolvem dados existentes na classe), além de efetuarem toda a comunicação com a classe *ReadData* (extraindo assim a informação desejada da mesma). De seguida são apresentados os métodos existentes:

- **Requisição Principal**: Existe uma requisição principal à classe *ReadData* para esta efetuar a leitura do ficheiro, e um posterior cálculo do número de páginas que os dados extraídos do ficheiro totalizam. Através deste método, a classe *PersonData* recebe o **total de páginas**, a **lista de *DataSeries*** criadas a partir dos dados importados e uma ***String JSON*** pronta a ser lida pela ferramenta de gráficos. Caso o total de páginas seja superior a duas, a classe *PersonData* envia imediatamente um pedido à classe *ReadData* para que esta crie uma *String JSON* com os dados correspondentes à primeira página de quinhentas medidas (ver secção 5.2.2.2). Evita-se assim automaticamente uma eventual sobrecarga da ferramenta de gráficos.
- **Requisição de Seguinte Intervalo de Medidas**: De cada vez que a Página HTML necessita do seguinte intervalo de dados para visualização (automaticamente ou por indicação do utilizador), a classe *PersonData* requere à classe ReadData esse mesmo pedido, indicando-lhe a lista de *DataSeries* original (obtida inicialmente pelo método de Requisição Principal) e qual a página que se deseja visualizar. É obtida uma *String JSON* com os dados correspondentes ao intervalo de tempo indicado, sendo esta disponibilizada para que a ferramenta de gráficos possa expor visualmente os dados. O método utiliza a informação sobre a página atual para saber qual a nova página a obter. Caso a página atualmente em visualização já seja a última, utilizar este método não efetuará nenhuma ação.
- **Requisição de Anterior Intervalo de Medidas**: Em tudo semelhante ao método para obtenção do Seguinte Intervalo de Medida, mas para obtenção do intervalo anterior ao

que se encontra em visualização no momento. Mais uma vez o intervalo de dados é assim disponibilizado para que a ferramenta de gráficos o posso expor. Caso a página atualmente em visualização já seja a primeira, utilizar este método não efetuará nenhuma ação.

- **Requisição de Visualização Total das Medidas:** Caso o utilizador esteja a visualizar um intervalo de medidas e deseje visualizar a sua totalidade, este é o método que permite tal efeito. Tal como os dois métodos anteriormente descritos, o método utiliza a Lista de *DataSeries* obtida inicialmente pelo método de Requisição Principal para requerer à classe *ReadData* que esta crie uma *String JSON* pronta a ser utilizada pela **ferramenta de gráficos** e que contenha todos os dados existentes na lista (e consequentemente no ficheiro lido). No entanto, a utilização deste método está condicionada pelo tamanho total dos dados a serem lidos. Se o total de intervalos de medidas (na prática, de páginas a ser visualizadas) for superior a 6, o método não efetua nenhuma ação, para prevenir um mau funcionamento da ferramenta de gráficos.
- **Requisição de Lista de Pastas:** Quando o utilizador deseja visualizar que pastas existem na pasta raiz do programa, este método é chamado. O método faz a requisição à classe *ReadData* para que esta preencha uma lista com todas as pastas existentes na pasta raiz e devolve-a à classe *PersonData*. Posteriormente a lista recebida é convertida num **mapa de pastas**, para que depois a Página HTML faça a apresentação das pastas disponíveis e o utilizador possa selecionar a desejada. Deste modo a classe *PersonData* saberá qual a pasta escolhida, informação que é fundamental para a seleção do ficheiro a visualizar.
- **Requisição de Lista de Ficheiros:** Selecionada a pasta da qual se pretende escolher um ficheiro (caso nenhuma esteja escolhida, por defeito é considerada a pasta raiz como escolhida), este método é o responsável por obter e disponibilizar um mapa de ficheiros passiveis de serem selecionados. O método faz uma requisição à classe *ReadData* para que esta (recebendo informação relativa à pasta onde o deve fazer) preencha uma lista de ficheiros com os existentes na pasta escolhida. Quando esta lista é devolvida à classe *PersonData*, a mesma converte-a num **mapa de ficheiros** para que a Página HTML possa apresentar as possibilidades de escolha ao utilizador, e quando este escolher uma, a classe *PersonData* armazene essa informação.
- **Restantes métodos:** Os restantes métodos são os *Getters* e *Setters*, utilizado para que a Página HTML tenha acesso às variáveis armazenadas na classe *PersonData*:
    - Os ***Getters*** permitem à Página HTML obter informação guardada na classe *PersonData*. A título de exemplo, existe um *Getter* para que a Página HTML possa saber o total de páginas a serem apresentadas e existe outro *Getter* para que a Página HTML aceda aos dados a ser enviados para a ferramenta de gráficos.
    - Os ***Setters*** permitem à Página HTML modificar variáveis guardadas na classe *PersonData*. Deste modo a página pode, por exemplo, definir qual a pasta que se deseja ler em busca de ficheiros e definir o ficheiro a ser lido. Esta informação está contida na classe *PersonData* para que a página depois lhe possa aceder e utilizar nos diversos métodos de requisição que possui.

5.2.3.2 Interface Gráfica - Página HTML

A parte final da ferramenta permite que o utilizador interaja com os recursos disponíveis, obtendo assim o desempenho desejado, visualizando os dados armazenados previamente. Apesar de ser apenas a ponta final da ferramenta, reveste-se de grande importância pois possibilita a análise qualitativa dos dados biomédicos armazenados. Resumindo, é necessário uma otimização desta etapa do processo para que o utilizador consiga tirar conclusões de valor sobre toda a informação coletada.

Para finalizar a ferramenta recorreu-se a duas Páginas HTML que apresentam ao utilizador um conjunto de opções e acesso à **ferramenta de gráficos**. Esta consiste num JavaScript componente de uma das páginas HTML que a partir de uma *String JSON* com os dados a ser visualizados cria um gráfico com toda a informação. Passa-se de seguida à explicação funcional destes componentes.

5.2.3.2.1 Página HTML Home

A primeira interação com o utilizador ocorre com uma Página HTML designada por Home. Esta página tem como objetivo selecionar o ficheiro a ser utilizado. Para tal, apresenta 3 botões e dois menus:

- **Botões**:
  - **Atualizar Lista de Pastas**: este botão ativa o método **Requisição de Lista de Pastas** da classe *PersonData*, que preenche o **menu Lista de Pastas** com todas as pastas passiveis de serem selecionadas;
  - **Atualizar Lista de Ficheiros**: este botão ativa o método **Requisição de Lista de Ficheiros** da classe *PersonData*, que preenche o **menu Lista de Ficheiros** com todos os ficheiros passiveis de serem selecionadas;
  - **Prosseguir**: procede para a seguinte Página HTML, a Página HTML Main.
- **Menus**
  - **Lista de Pastas**: lista de pastas que podem ser escolhidas. Com base na selecionada, a lista de ficheiros existente nessa mesma pasta será preenchida;
  - **Lista de Ficheiros**: lista de ficheiros existentes na pasta selecionada. O ficheiro selecionado será utilizado na **Página HTML Main** e consequentemente pela ferramenta de gráficos.

5.2.3.2.2 Página HTML Main

Depois de passar pela Página Home, selecionar o ficheiro a ser utilizado e clicar no botão *Proceed* (inglês para Prosseguir), o utilizador depara-se com a **Página Principal**, com campos de texto, vários botões e com espaço reservado à **ferramenta de gráficos**. De seguida são apresentados os componentes da Página Principal:

- **Caixas de Texto**
  - **Título do Ficheiro**: tal como o nome indica, uma caixa de texto é apresentada indicando ao utilizador qual o ficheiro selecionado;
  - **Página**: apresentada a página de dados atualmente em visualização, em função do total de páginas extraídas do ficheiro;
  - **IMEI:** apresenta o IMEI do dispositivo que coletou os dados a partir dos sensores e os enviou para o servidor;
  - **Data:** apresenta a **data de início da monitorização** do ficheiro em visualização**.**

Todas as informações contidas nas caixas de texto são extraídas da classe *PersonData* através dos referidos *Getters*.

- **Botões:**
  - **Atualizar (Refresh):** Botão fundamental ao funcionamento da ferramenta de gráficos. Clicando neste botão é ativado o método de **Requisição Principal** na classe *PersonData*. Apenas neste momento os dados são importados do ficheiro para o programa, ficando disponíveis para visualização e manipulação.
  - **Página Anterior (Previous Page): E**ste botão chama o método **Requisição de Anterior Intervalo de Medidas** de modo a obter o intervalo de medidas anterior

ao que se encontra em visualização (exemplo: se a página atual é a 3, pede o intervalo correspondente à página 2).
    - **Próxima Página (Next Page):** Este botão chama o método **Requisição de Seguinte Intervalo de Medidas** de modo a obter o intervalo de medidas imediatamente a seguir ao que se encontra em visualização (exemplo: se a página atual é a 3, pede o intervalo correspondente à página 4).
    - **Mostrar Todos os Dados (Show All):** Botão cujo objetivo é, em caso de possibilidade técnica, exibir todos os dados extraídos do ficheiro. Apenas funciona caso o total de dados da maior *DataSeries* criada a partir do ficheiro não seja superior a 6.
    - **Visualizar:** Ativa a Ferramenta de Gráficos, o JavaScript, indicando-lhe a localização dos dados a ser visualizados. Após clicar neste botão, os dados são visualizados num gráfico que se apresenta na página, em espaço especialmente reservado ao JavaScript.
    - **Início:** Permite voltar à Página HTML de Login para escolher um novo ficheiro.
- **JavaScript - Ferramenta de Gráficos:** O elemento fundamental da visualização, a Ferramenta de Gráficos consiste num JavaScript que recebe uma *String JSON* com as *DataSeries* a serem visualizadas (com o **título**, **unidades** e valores das **medidas** presentes em cada uma). Este JavaScript recebe a informação e desenha um gráfico que expõe as medidas recebidas, em função da data de aquisição de cada uma. O título do gráfico e título dos eixos podem ser personalizados, bem como os símbolos utilizados para apresentação dos dados e o fundo do gráfico (entre outras opções). Existem dois botões no canto superior direito que permitem a impressão e/ou exportação dos gráficos obtidos. Esta componente foi baseada numa ferramenta disponível para *download* se para uso pessoal, a highcharts [50] .

Entre as características do gráfico criado pelo JavaScript destacam-se as seguintes:

- O eixo das abcissas possui uma escala temporal;
- O título das *DataSeries* aparece na legenda. É possível ocultar *DataSeries* ao clicar no seu nome na legenda. Caso se utilize esta função, a escala vertical do gráfico ajusta-se automaticamente aos dados em visualização no momento.
- Ao passar com o ícone do rato ao longo do gráfico é possível obter informação extra sobre cada ponto do gráfico (a sua **data de aquisição** exata, as **unidades** e o **valor** exato).

Todas estas características estão detalhadas no próximo capítulo, referente a Resultados e Discussão. Está assim finalizada a descrição funcional de toda a Ferramenta de Visualização de Dados.

6 Resultados e Discussão

Neste capítulo serão apresentados os resultados obtidos no decorrer do projeto, bem como a discussão da validade e qualidade dos mesmos. Pretende-se averiguar se o projeto foi bem-sucedido e se de facto constitui uma mais-valia tecnológica para o produto OneCare e a Engenharia Biomédica. Os resultados serão divididos em duas partes correspondentes às duas partes descritas na Metodologia.

6.1 Programa de Receção de Armazenamento de Dados

Na primeira parte do trabalho, o objetivo passou por garantir que todos os dados enviados pelos sensores eram devidamente recebidos, tratados e armazenados. Para tal foram feitos vários testes ao sistema, que permitiram melhorar o programa até ao estado atual do mesmo. Estes testes consistiram essencialmente no envio de dados conhecidos para o servidor e comparação com os dados recebidos, em diferentes circunstâncias que visam recriar o ambiente real de utilização da tecnologia desenvolvida. Neste capítulo serão apresentados os testes principais que garantiram o bom funcionamento do programa, bem como resultados obtidos aquando de situações reais de utilização do programa.

6.1.1 Teste de Receção de Dados

O primeiro teste efetuado visou averiguar se o cliente recebia todos os dados enviados, procurando garantir a fiabilidade do programa. Os dados foram enviados por um cliente MQTT que publicava no mesmo tópico em que o programa que os recebe estava subscrito. O resultado é apresentado no Anexo A com o título **Resultados de Teste de Receção de Dados**.

O teste corresponde a uma medição de um minuto de Oximetria de Pulso (tipo um, visível na mensagem). As mensagens foram enviadas de segundo a segundo (o intervalo de aquisição varia de mil em mil milissegundos), seguindo o expectável pois a taxa de aquisição é de um valor por segundo (1Hz). É possível também observar o IMEI do dispositivo que enviou os dados (658g334n1a8hhk8 no exemplo do Anexo A), bem como o *timestamp* de início de monitorização que pode ser utilizado para obter a data correspondente.

Como é possível comprovar, todos os dados foram recebidos. Ao longo do projeto foram recebidos milhares de dados em que o programa se comportou sempre irrepreensivelmente, mesmo quando **recebia várias mensagens por segundo**. A fiabilidade de receção de dados ficou assim garantida.

6.1.2 Teste de Intervalo decorrido entre Envio e Receção de Mensagens

Na sequência do teste anterior, o próximo visou estimar o intervalo temporal decorrido entre o envio dos dados para o servidor e a sua receção pelo programa. O programa final não irá receber as **datas de aquisição** como apresentadas de seguida, mas sim como o intervalo entre a **data de início da monitorização** e o **momento real** da medição (ver secção 5.1.3 a referência ao campo *AQUISITION TIME GAP*).

Para este teste foi feita essa pequena alteração com o objetivo de comparar os tempos de envio dos dados e de receção mais facilmente. Os resultados obtidos estão apresentados no Anexo A com título **Teste de Intervalo Temporal entre Envio e Entrega de Dados**.

A observação das imagens, em particular dos campos *aq_stamp* (*timestamp* correspondente ao momento em que a mensagem foi enviada para o servidor, em milissegundos) e do último valor de cada linha (*timestamp* que corresponde ao momento em que a mensagem foi recebida pelo programa), permite tirar ilações acerca do intervalo decorrido entre estes dois momentos. Os

dois valores foram comparados utilizando uma folha de cálculo, tendo sido obtidos os resultados apresentados na Tabela 3.

Tabela 3: Análise de intervalo decorrido entre envio e receção de mensagens

Média de tempo demorado entre envio de mensagem e sua receção pelo programa	93 Milissegundos
Máximo de tempo demorado entre envio de mensagem e sua receção pelo programa	161 Milissegundos
Mínimo de tempo demorado entre envio de mensagem e sua receção pelo programa	49 Milissegundos

Tanto o cliente que publicou as mensagens, como o cliente que as recebeu, estavam ligados a uma rede *wireless* estável. No entanto é de crer que este facto não tem importância significativa para a discussão, devido ao cariz do protocolo MQTT (apropriado a redes de baixa largura de banda) e ao pequeno tamanho das mensagens.

Através destes testes foi possível verificar que o tempo de transmissão de mensagens entre a fonte das mesmas e o programa é muito reduzido. O valor médio foi de 93 milissegundos o que traduz bem este facto. Além deste valor médio, o valor máximo foi de 161 milissegundos e o valor mínimo foi de 49 milissegundos. Estes dados apenas comprovam as conclusões que experiência de utilização do programa permitiu tirar: o tempo de transmissão de mensagens enviadas do dispositivo que colecta os dados biomédicos e o programa é muito curto.

6.1.3 Teste de Receção de Mensagens enviadas aquando do programa desconectado do *broker*

O teste final de receção de mensagens consiste na verificação da capacidade do programa poder receber dados que tenham sido enviados para o *broker* instalado no servidor sem que o programa estivesse conectado.

O desejado é que quando o programa re-conecta, os dados enviados entretanto sejam todos recebidos. Para tal, o cliente MQTT serve-se de algumas propriedades do *broker* utilizado e bem como de propriedades do MQTT (*QoS* superior a 0). Para demonstrar que o programa tem esta capacidade foi feita uma modificação temporária ao mesmo, que assim expõe a data de chegada da mensagem para permitir comparar esta com a data original aquisição. A situação de teste foi criada desconectando o cliente do servidor, enviando dados, e re-conectando o cliente ao servidor passado vários minutos.

No Anexo A está o resultado de um destes testes (com o título **Teste de Entrega de Dados após Re-conexão do Broker**). Este permite comprovar que todos os dados foram recebidos – 60 dados correspondentes a um minuto de monitorização de Oximetria de Pulso, a uma taxa de aquisição de uma medida por segundo. Os dados foram recebidos cerca de 48 minutos depois de terem sido enviados, tempo que demorou a ligar o cliente. Numa situação normal, o cliente seria imediatamente reiniciado, mas para este teste resolveu deixar-se o cliente desconectado durante mais tempo do que numa situação real. Comparando os tempos de receção de mensagens, as mensagens correspondentes aos sessenta valores enviados foram recebida todas em cerca de 1 segundo (exatamente 1,094 segundos) o que demonstra bem a capacidade de receção de dados pelo programa.

Ao longo da utilização do programa esta situação foi repetida várias vezes (por vezes intencionalmente, outras vezes acidentalmente) e o programa recebeu sempre todas as mensagens enviadas. Um detalhe importante para utilizar esta funcionalidade é o *broker* ter obrigatoriamente que estar configurado para tal efeito.

Este foi o último teste às capacidades de receção de mensagens por parte do cliente MQTT do programa. Através destes testes foi possível garantir que o programa apresenta as condições necessárias à sua utilização para receber dados biomédicos, cuja importância requere um grau de fiabilidade de funcionamento elevado.

É assim garantido que os dados são recebidos sempre que enviados por uma fonte de dados, sem adulterações, lacunas ou falhas de comunicação. Pretende-se de seguida garantir que as mensagens são devidamente descodificadas e a informação nelas contida é armazenada.

6.1.4 Escrita de Ficheiros

Depois de testada a capacidade de receção de mensagens pelo programa, será demonstrada a composição do ficheiro temporário que este escreve com os dados biomédicos recebidos. Como descrito na secção 5.1.3, o programa vai escrevendo os dados extraídos das mensagens recebidas num ficheiro temporário que posteriormente será verificado para garantir que os dados estão organizados temporalmente. A descodificação da mensagem *JSON* não possui nenhum *output* visível por si só, mas o resultado desta descodificação é escrito num ficheiro, pelo que um ficheiro corretamente elaborado significa que a descodificação foi executada corretamente. Segue um exemplo deste tipo ficheiro onde se pode comprovar como os dados são organizados. A Figura 27 corresponde a parte do ficheiro aberto num editor simples de texto. Mais ficheiros podem ser encontrados no Anexo C.

658g334n1a8hhk8_1378649810531.csv

```
1  "IMEI","658g334n1a8hhk8"
2  "INITIAL TIME","1378649810531"
3  "TYPE","AQUISITION TIME GAP","AQUISITION RATE","VALUE"
4  "1","0","1","98"
5  "1","1000","1","98"
6  "1","2000","1","98"
7  "1","3000","1","98"
8  "1","4000","1","98"
9  "1","5000","1","98"
10 "1","6000","1","98"
11 "1","7000","1","98"
12 "1","8000","1","98"
13 "1","9000","1","98"
14 "1","10000","1","98"
15 "1","11000","1","98"
16 "1","12000","1","98"
```

Figura 27: Ficheiro de dados, aberto em bloco de notas

A Figura 28 representa a mesma parte mas aberta numa folha de cálculo (*Microsoft Excel 2013*). Como se pode observar, é possível abrir os dados em *software* apropriado, o que pode ser útil numa futura utilização dos ficheiros.

Com estes exemplos pretende-se apresentar os ficheiros temporários escritos pelo programa. É possível observar as duas linhas de cabeçalho identificativo (com IMEI e data de inicio de monitorização, bem como os títulos dos campos armazenados) e a informação propriamente dita, nas linhas seguintes. Todos os dados são armazenados desta forma, que garante fiabilidade de armazenamento e leitura, ao mesmo tempo que proporciona o uso de ficheiros pequenos e fáceis de ler. A escrita para ficheiro é feita imediatamente após cada mensagem ser recebida, ficando o ficheiro disponível logo após a sua escrita.

	A	B	C	D
1	IMEI	658g334n1a8hhk8		
2	INITIAL TIME	1378649810531		
3	TYPE	AQUISITION TIME GAP	AQUISITION RATE	VALUE
4	1	0	1	98
5	1	1000	1	98
6	1	2000	1	98
7	1	3000	1	98
8	1	4000	1	98
9	1	5000	1	98
10	1	6000	1	98
11	1	7000	1	98
12	1	8000	1	98
13	1	9000	1	98
14	1	10000	1	98
15	1	11000	1	98
16	1	12000	1	98

Figura 28: Ficheiro de dados, aberto em software de folha de cálculo

Os ficheiros finais serão semelhantes, apenas será garantida a ordem temporal dos dados para prevenir erros aquando da sua visualização. No Anexo C encontram-se vários exemplos de ficheiros escritos pelo programa, para serem consultados.

6.1.5 Organização e Escrita do Ficheiro Final

Os ficheiros finais não são mais do que os ficheiros temporários após ser garantida a sua organização temporal. Este passo é muito importante pois caso os ficheiros não sejam devidamente organizados, o JavaScript responsável pela sua visualização final não iria funcionar.

O programa possui a capacidade de ler os ficheiros temporário já finalizados, organizar toda a informação contida nos mesmos, e escrever um ficheiro final de estrutura e conteúdo similar ao original mas garantindo a correcta ordenação dos dados – por ordem crescente de data de aquisição. De seguida é apresentado um exemplo do que esta funcionalidade efetua. A Figura 29 ilustra um ficheiro cuja informação não está devidamente ordenanda.

É possível verificar que as linhas 5 e 9 estão trocadas, bem como as linhas 15 e 18. O que esta função programa faz é ler este ficheiro, ordenar internamente os dados e no final escrever os dados ordenados para um novo ficheiro, o ficheiro final, armazenado com os restantes ficheiros correspondentes a monitorizações do mesmo indivíduo.

Observando este exemplo é possível observar o que o programa tem a capacidade de fazer: organizar devidamente os dados. A utilização desta funcionalidade nunca se deparou com problemas, sendo que no final foram sempre obtidos ficheiros organizados.

	A	B	C	D
1	IMEI	658g334n1a8hhk8		
2	INITIAL TI	1378649810532		
3	TYPE	AQUISITION TIM	AQUISITIC	VALUE
4	1	0	1	98
5	1	5000	1	99
6	1	2000	1	98
7	1	3000	1	98
8	1	4000	1	98
9	1	1000	1	97
10	1	6000	1	98
11	1	7000	1	98
12	1	8000	1	98
13	1	9000	1	98
14	1	10000	1	98
15	1	14000	1	99
16	1	12000	1	98
17	1	13000	1	98
18	1	11000	1	97
19	1	15000	1	98

Figura 29: Ficheiro de dados, desorganizado

	A	B	C	D
1	IMEI	658g334n1a8hhk8		
2	INITIAL TIME	1378649810532		
3	TYPE	AQUISITION TIME GAP	AQUISITION RATE	VALUE
4	1	0	1	98
5	1	1000	1	97
6	1	2000	1	98
7	1	3000	1	98
8	1	4000	1	98
9	1	5000	1	99
10	1	6000	1	98
11	1	7000	1	98
12	1	8000	1	98
13	1	9000	1	98
14	1	10000	1	98
15	1	11000	1	97
16	1	12000	1	98
17	1	13000	1	98
18	1	14000	1	99
19	1	15000	1	98

Figura 30: Ficheiro de dados após organização

6.2 Ferramenta de Visualização de Dados

Depois de demonstrado o funcionamento do programa de receção e armazenamento de dados, serão agora apresentados os resultados do desenvolvimento da ferramenta de visualização de dados. Será dado especial enfase à interface gráfica, uma vez que todo o programa por detrás desta ferramenta tem como função preparar os dados para serem visualizados pelo programa.

6.2.1 Página HTML *Home*

O primeiro contacto do utilizador com a ferramenta é feita na Página *Home*, escrita em HTML.

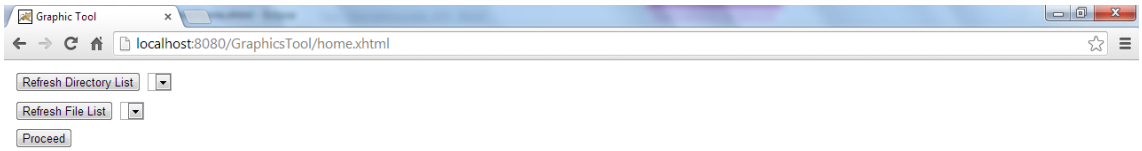


Figura 31: Página inicial HTML Home

Esta é a página que permite escolher o ficheiro a selecionar, bem como a pasta onde se deve procurar pelo ficheiro. A Figura 31 corresponde à página *Home* no seu estado inicial. Após clicar no botão *Refresh Directory List*, o menu em frente a este botão é preenchido com as pastas existentes, que permitem organizar os ficheiros (atualmente pelo IMEI do aparelho que enviou os dados). O resultado está apresentado na Figura 32:

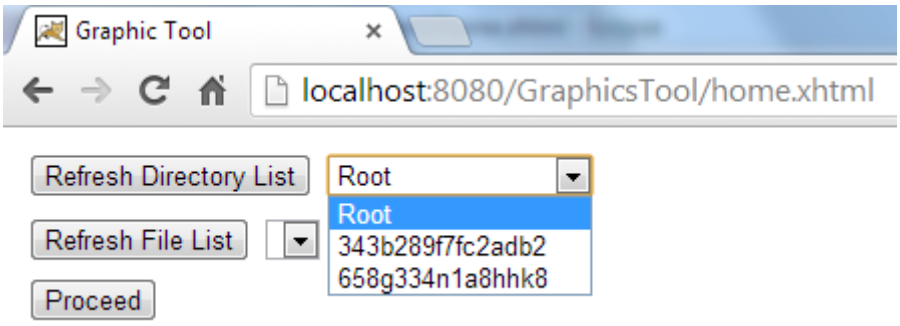


Figura 32: Seleção de pasta na Página Home

Após selecionar uma pasta, é possível obter a lista de ficheiros existentes nessa pasta. Clicando no botão *Refresh File List*, o menu em frente a este botão é preenchido com os ficheiros existentes na pasta selecionada.

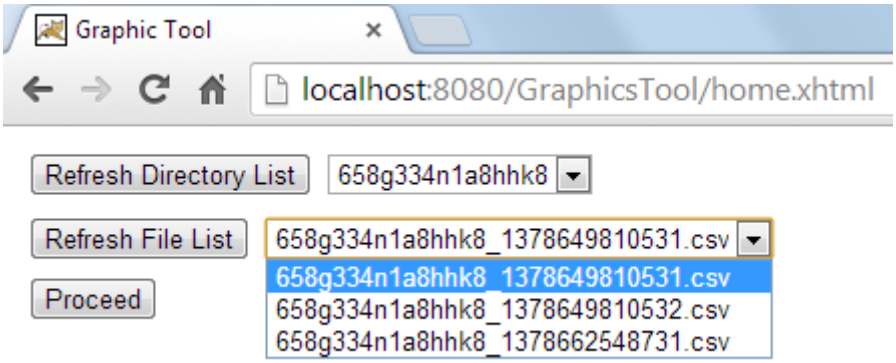


Figura 33: Seleção de ficheiro na Página Home

Depois de selecionar um ficheiro, o utilizador deve clicar no botão *Proceed* para prosseguir e aceder à página principal da ferramenta, onde poderá visualizar os dados armazenados e receberá mais informações sobre o ficheiro escolhido.

6.2.2 Página HTML Main

Depois de proceder para o ecrã principal (Main) da ferramenta, a Página HTML apresentada adquire a aparência ilustrada na Figura 34. Caso não seja selecionado nenhum ficheiro, a interface devolve um aviso representado na Figura 35, onde é visível um aviso de que nenhum ficheiro foi selecionado (em inglês, tal como o resto da interface).

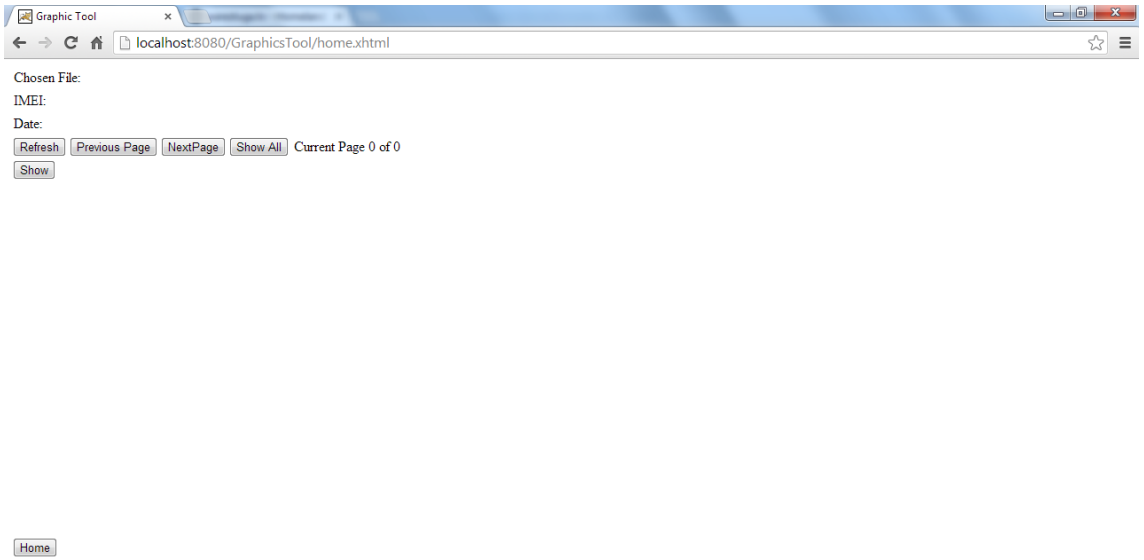


Figura 34: Página HTML Main vazia

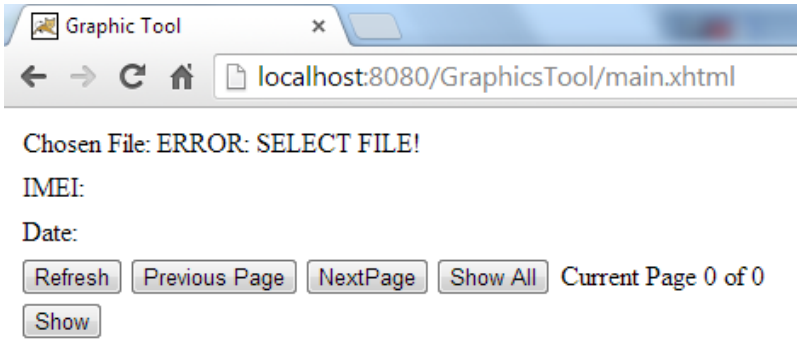


Figura 35: Mensagem de erro na Página Main por não ter sido selecionado qualquer ficheiro

Em caso da seleção ter sido devidamente efetuada, o ecrã apresenta imediatamente a seguinte informação:

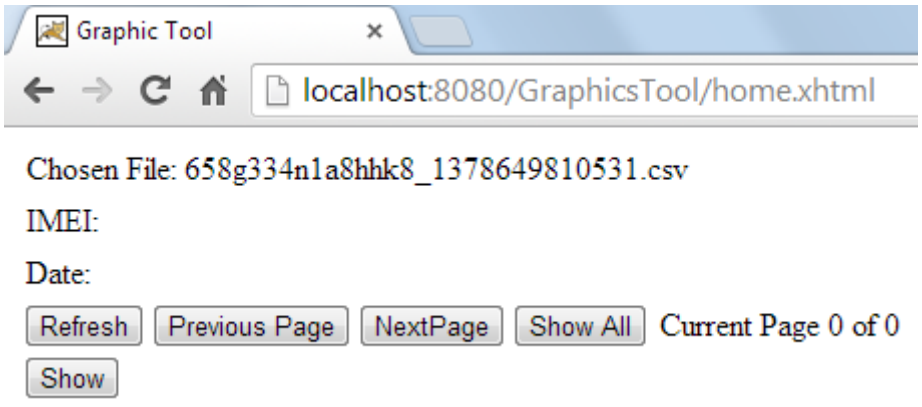


Figura 36: Página Main com o nome do ficheiro selecionado

Após ter a confirmação do ficheiro selecionado (em cima, no campo *Chosen File*), deve clicar-se no botão *Refresh* para que o programa por detrás da ferramenta possa importar e tratar os dados do ficheiro. Além da importação dos dados, serão preenchidos os campos *IMEI* e *Date* na Página HTML, bem como o total de páginas que os dados totalizam.

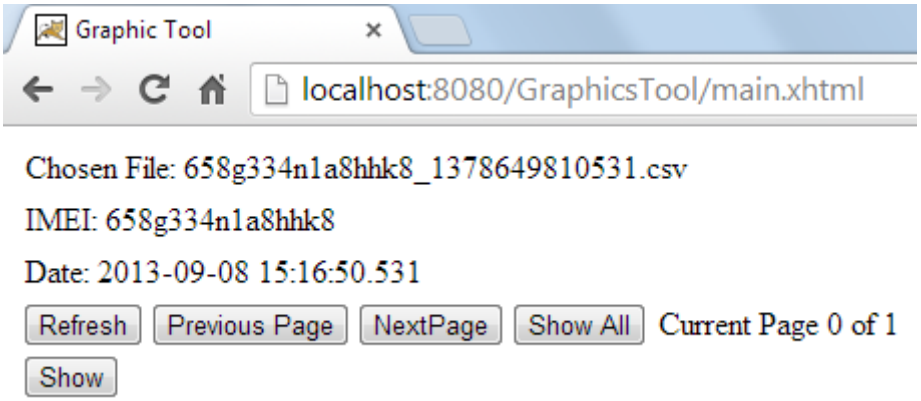


Figura 37: Informação relativa ao ficheiro selecionado na Página Main

Depois de visualizar este ecrã, basta clicar no botão *Show* para que a ferramenta apresente o gráfico correspondente aos dados importados do ficheiro. Neste exemplo, apresentado na Figura 38, apenas existe uma página para visualização, pelo que serão visualizados todos os dados disponíveis.

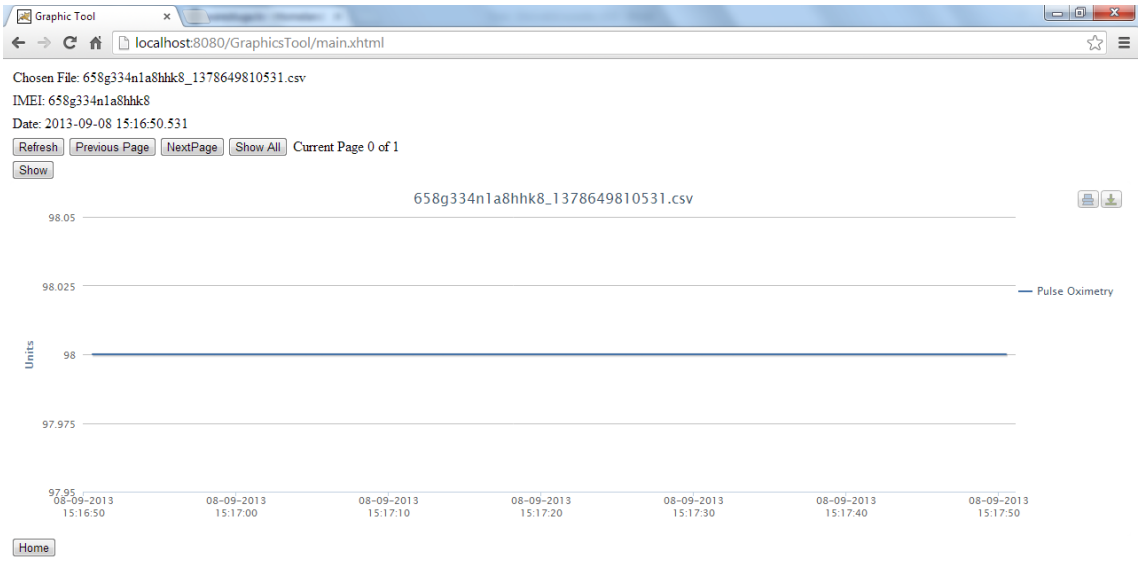


Figura 38: Página Main a exibir gráfico de apenas uma série de dados

Este gráfico correspondeu a medida de Oximetria de Pulso durante um minuto, pelo que não existiram variações no gráfico. Na Figura 39 é apresentado um gráfico que contém dados correspondentes a duas *DataSeries* adquiridas na mesma monitorização: uma de Oximetria de Pulso e outra de Frequência Cardíaca.

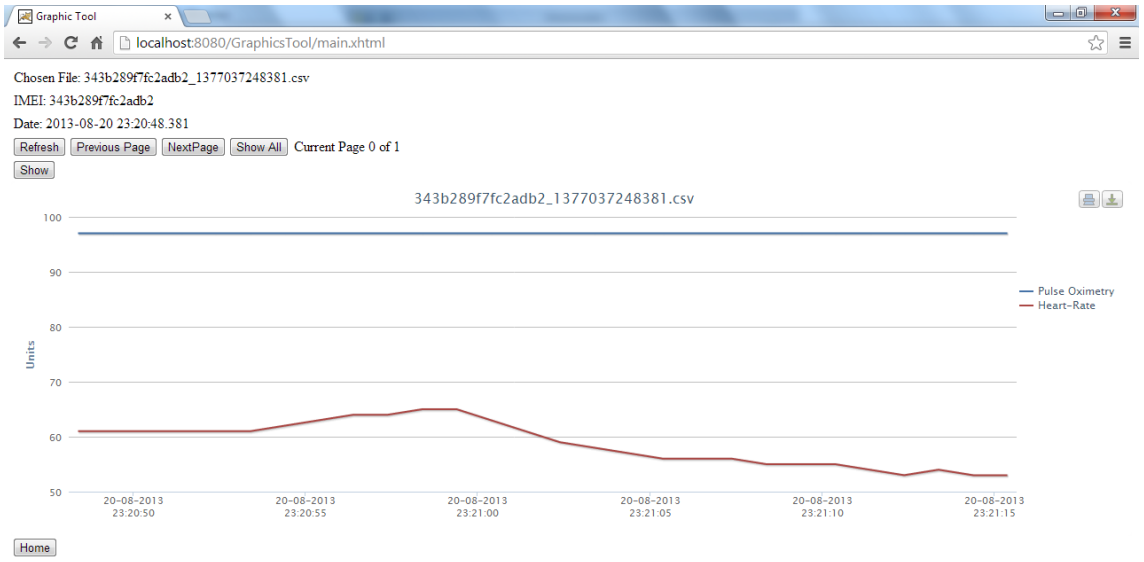


Figura 39: Página Main a exibir um gráfico de duas séries de dados

Este gráfico permite demonstrar outras duas capacidades do JavaScript utilizado:

- Ao passar o cursor pelo gráfico, são realçados os pontos do mesmo e exposta informação como o valor efetivo do ponto e as unidades do mesmo:

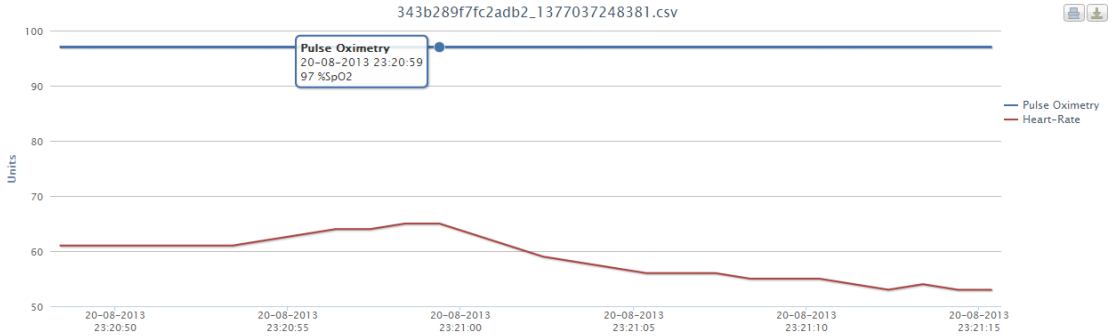


Figura 40: Detalhe de gráfico de dados

- A possibilidade de esconder séries de dados, de modo a focar a escala do gráfico nas séries de dados desejadas;

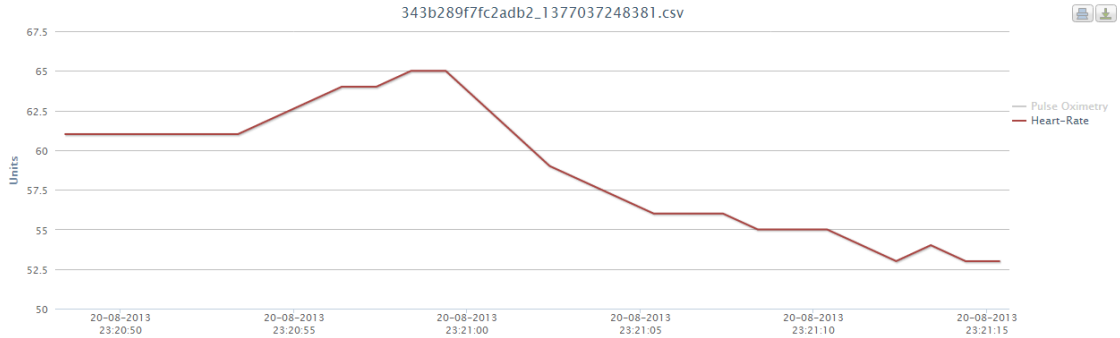


Figura 41: Seleção de apenas uma série de dados num gráfico com várias

Se não for selecionada nenhuma *DataSeries*, o gráfico apresenta-se vazio.

De seguida é apresentado um gráfico de Eletrocardiograma que além de ser mais rico em variações, irá ser dividido em intervalos. A Figura 42 corresponde ao gráfico obtido

automaticamente pela ferramenta: como o total de páginas é superior a duas, a ferramenta efetua uma divisão por intervalos e apresenta apenas a primeira página de dados.

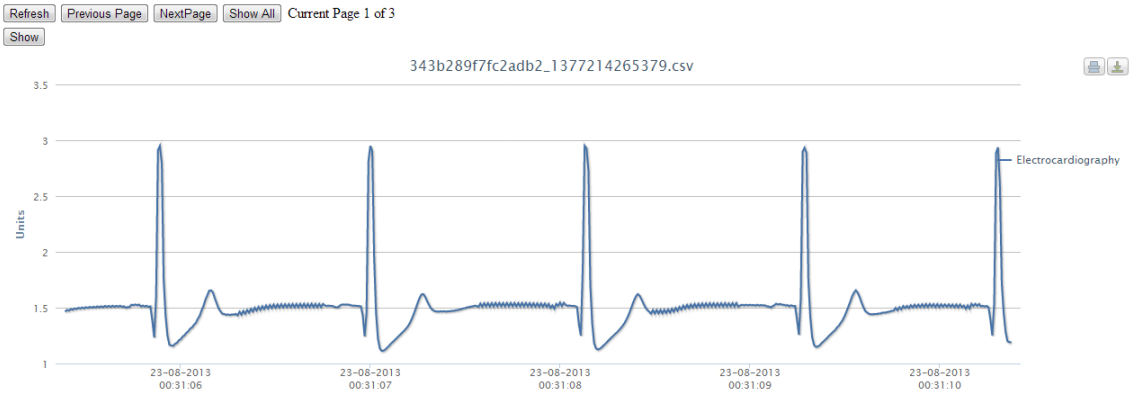


Figura 42: Gráfico de ECG desenhado pela Ferramenta de Gráficos - Intervalo

Clicando nos botões *Previous Page* (Página Anterior) e *Next Page* (Próxima Página) é possível navegar entre páginas de dados. Selecionado o intervalo de dados a visualizar, basta apenas clicar no botão *Show* para que seja apresentado o gráfico correspondente.

Utilizando o botão *Show All*, todos os dados recebidos importados do ficheiro desta monitorização são visualizados, desde que totalizem menos de seis páginas (para não sobrecarregar a ferramenta). E possível verificar a forma típica do ECG neste gráfico, apesar de serem visíveis algumas anomalias, a que o programa é alheio pois provêm da fonte original dos dados, os sensores utilizados.

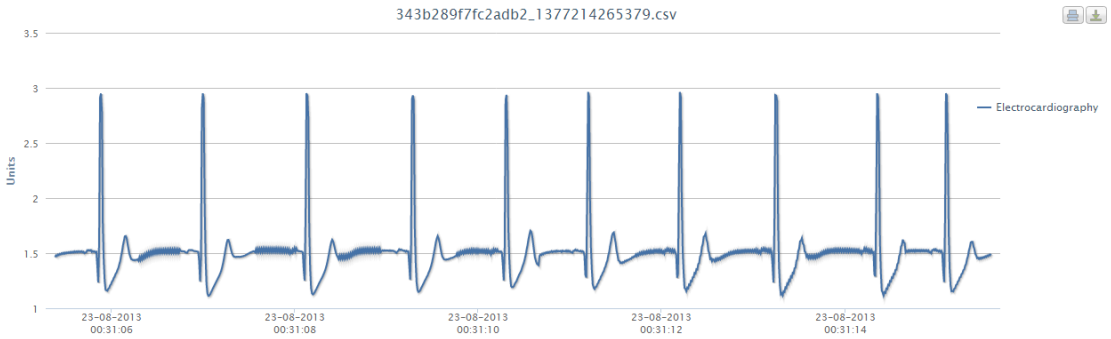


Figura 43: Gráfico de ECG desenhado pela Ferramenta de Gráficos - Total

A última funcionalidade a mencionar é a possibilidade de imprimir ou exportar o gráfico, utilizando os dois botões do canto superior direito do JavaScript. Através do primeiro botão, é possível imprimir o gráfico obtido diretamente a partir da ferramenta (Figura 44).

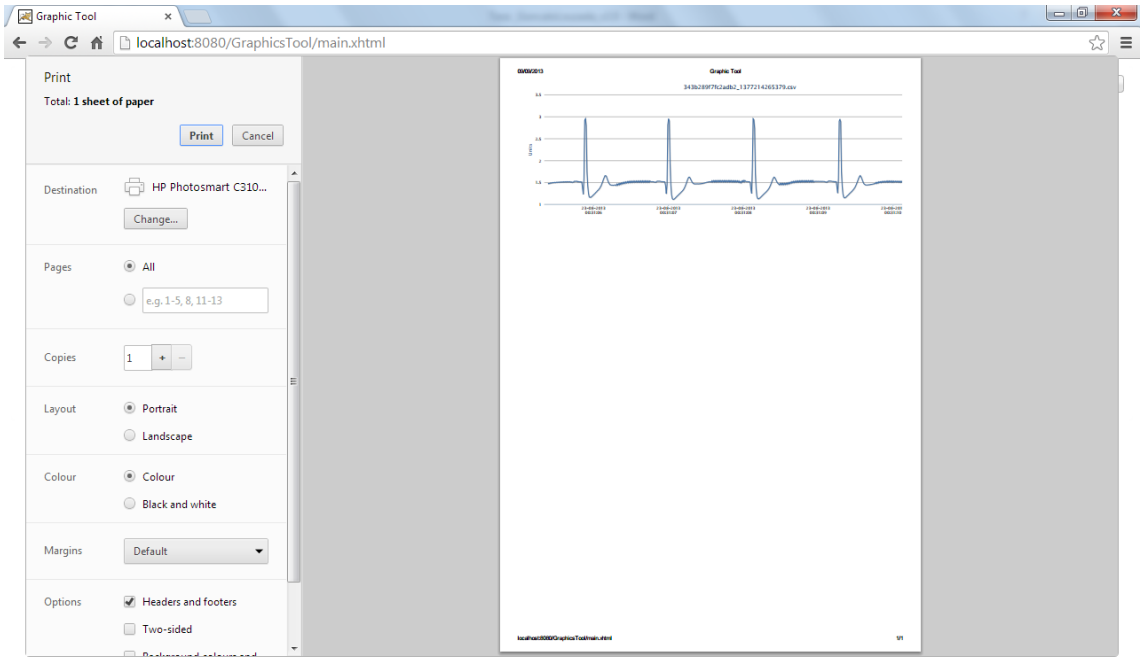


Figura 44: Ecrã de impressão de gráfico de dados

Clicando no segundo botão surgem 4 opções para o utilizador escolher em que formato deseja exportar o gráfico: imagem PNG, imagem JPEG, documento PDF ou imagem vetorial SVG.

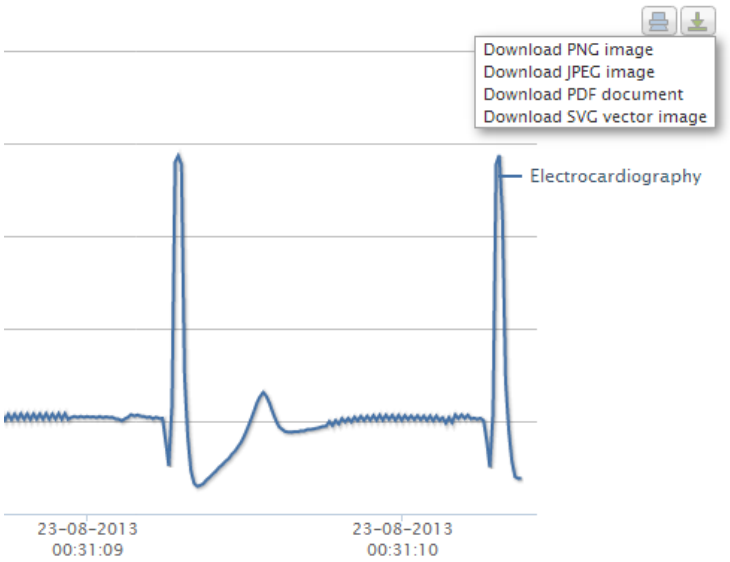


Figura 45: Ecrã de exportação no JavaScript

Todas as opções foram testadas, seguindo um exemplo de uma imagem PNG obtida através da opção de exportação na Figura 46.

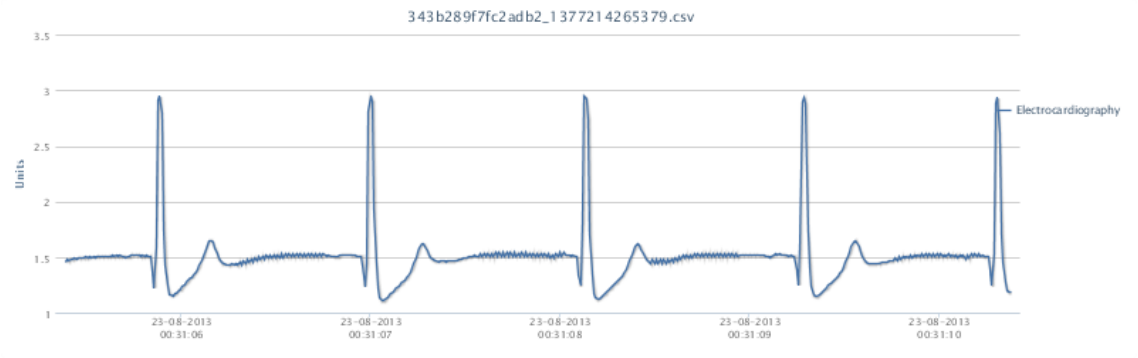


Figura 46: Imagem PNG exportada pelo JavaScript

Como se pode verificar, o intervalo em visualização foi exportado para uma imagem correspondente.

6.3 Considerações Finais

Em ambas as partes do projeto, os testes realizados permitiram chegar a resultados positivos, que levam a crer que o trabalho foi bem desenvolvido.

O programa de receção de dados cumpriu sempre as suas funções sem qualquer problema apresentado, mesmo quando sujeito a grandes quantidades de mensagens em curtos espaços de tempo (várias mensagens por segundo). Os dados foram sempre corretamente armazenados.

A ferramenta de visualização também cumpriu com aquilo que era esperado, permitindo uma seleção e visualização intuitiva de dados, que num contexto médico pode ser de grande utilidade. O desempenho da ferramenta foi sempre eficaz, não tendo tempos de espera e permitindo desenhado gráficos nítidos e de clara interpretação.

7 Conclusão

O projeto apresentado consiste no projeto final de Mestrado Integrado em Engenharia Biomédica e foi desenvolvido ao longo do ano letivo 2012/2013, no âmbito do produto OneCare da Intellicare. O resultado do mesmo será incorporado neste produto, e tem como objetivo funcional a receção, armazenamento e visualização de sinais biomédicos remotamente. Os dados são enviados através da Internet para os programas desenvolvidos no projeto, que tratam de os armazenar e disponibilizar para visualização. Deste modo é possível aceder a dados de uma monitorização de saúde remotamente, bastando que para isso os dados tenham sido enviados para o programa. Um médico, prestador de cuidados ou familiar do utilizador podem assim aceder em qualquer altura aos dados.

O projeto serviu-se de vária tecnologias, entre as quais as principais foram a linguagem de programação Java, o protocolo de comunicação MQTT, a linguagem de programação interpretada JavaScript, entre outras das que mais recentes avanços têm sofrido. Procurou-se deste modo garantir que o projeto se encontra na vanguarda da tecnologia.

O programa de receção de dados revelou-se fiável, não tendo falhado nenhuma mensagem que lhe tenha sido enviada. As mensagens recebidas foram corretamente desencriptadas e armazenadas em ficheiros. Os ficheiros temporários foram sempre gerados corretamente, e imediatamente após a receção das mensagens. O tempo de realização destas tarefas foi extremamente reduzido, possibilitando a disponibilização de um ficheiro final, organizado, segundos após ter sido finalizada a monitorização. Tal comprova o bom funcionamento do programa, que além de fiável se revelou muito rápido no cumprir das suas funções, o que neste tipo de produtos é muito importante. O facto de ser muito rápido a receber e escrever ficheiros permite uma monitorização em tempo quase real, o que se traduz numa mais-valia na monitorização de pacientes, diminuindo o tempo de intervenção em caso de necessidade.

A ferramenta de visualização gráfica de dados também se revelou um sucesso no que toca aos objetivos propostos. Os ficheiros armazenados são intuitivamente selecionados e corretamente importados. A informação é apresentada com clareza, e os gráficos são elucidativos dos dados armazenados. Com uma fonte de dados adequada, esta ferramenta pode assim ser utilizada num contexto biomédico, permitindo uma análise concisa dos dados captados e assim retirar conclusões acerca dos mesmos. Um médico ou prestador de cuidados pode assim extrair informação importante relativa ao estado de saúde de um utilizador dos sensores que enviem dados para esta ferramenta.

No final do projeto é possível afirmar que o trabalho desenvolvido é um acréscimo de valor à solução OneCare, providenciando novas ferramentas que assim enriquecem a solução existente e a tornam mais completa e atual. Os objetivos foram cumpridos e os programas e ferramentas desenvolvidos estão prontos para utilização.

7.1 Trabalho Futuro

Apesar de terminado, existem ainda soluções que podem ser adicionadas ao trabalho desenvolvido.

A aplicação e teste em massa dos programas criados pode revelar oportunidades de melhoria que com os testes realizados não tenham sido captados. Esta é uma etapa que deve ser cumprida no futuro, pois só assim poderão ser encontrados pontos de melhoria quer do programa de receção e armazenamento, quer da ferramenta de visualização.

7.2 Publicações Associadas

No Anexo D pode ser encontrado um artigo, aprovado para a *International Conference on Health and Social Care Information Systems and Technologies* a ser realizada entre 23 e 25 de Outubro de 2013 em Lisboa, e elaborado no âmbito deste projeto.

Referências

[1] "PORDATA: Esperança de vida à nascença: total e por sexo - Portugal," [Online]. Available: http://www.pordata.pt/Portugal/Esperanca+de+vida+a+nascenca+total+e+por+sexo-418. [Acedido em 12 Novembro 2012].

[2] "PORDATA: Esperança de vida aos 65 anos: total e por sexo - Portugal," [Online]. Available: http://www.pordata.pt/Portugal/Esperanca+de+vida+aos+65+anos+total+e+por+sexo-419. [Acedido em 12 Novembro 2012].

[3] "PORDATA: Óbitos de residentes em Portugal: total e no primeiro ano de vida - Portugal," [Online]. Available: http://www.pordata.pt/Portugal/Obitos+de+residentes+em+Portugal+total+e+no+primeiro+ano+de+vida-15. [Acedido em 12 Novembro 2012].

[4] "PORDATA: População residente segundo os Censos: total e por sexo - Portugal," [Online]. Available: http://www.pordata.pt/Portugal/Populacao+residente+segundo+os+Censos+total+e+por+sexo-1. [Acedido em 12 Novembro 2012].

[5] "PORDATA: Óbitos por algumas causas de morte em Portugal (percentagem)," [Online]. Available: http://www.pordata.pt/Portugal/Obitos+por+algumas+causas+de+morte+(percentagem)-758. [Acedido em 12 Novembro 2012].

[6] "ISA - Intelligent Sensing Anywhere, S.A.," [Online]. Available: http://www.isasensing.com/. [Acedido em 1 Setembro 2013].

[7] "Wikipedia - Circulatory System," [Online]. Available: http://en.wikipedia.org/wiki/Circulatory_system. [Acedido em 22 Janeiro 2013].

[8] "Wikipedia - Human Heart," [Online]. Available: http://en.wikipedia.org/wiki/Human_heart. [Acedido em 14 Novembro 2012].

[9] J. E. Hall, Textbook of Medical Physiology, 2005.

[10] J. Hampton, The ECG Made Easy, 6ª ed., Churchill Livingstone, 2006.

[11] J. D. Trigo, A. Iglesias, I. Martínez e J. García, "A Review on Digital ECG Formats and the Relationships Between Them.," *IEEE Transactions on Information Technology in Biomedicine,* vol. 16, pp. 432-444, Maio 2012.

[12] J. Webster, Design of Pulse Oximeters, Institute of Physics Pub., 1997.

[13] J. Moyle, Pulse Oximetry, 2ª ed., Londres: BMJ Books, 2002.

[14] "Learn About Java Technology," [Online]. Available: http://www.java.com/en/about/. [Acedido em 12 Setembro 2013].

[15] "Wikipedia - Integrated Development Environment," [Online]. Available: http://en.wikipedia.org/wiki/Integrated_development_environment. [Acedido em 2013 Setembro 1].

[16] "JSON," [Online]. Available: http://www.json.org/. [Acedido em 6 Setembro 2013].

[17] V. Lampkin, "What is MQTT and how does it work with WebSphere MQ?," 18 Abril 2013. [Online]. Available: https://www.ibm.com/developerworks/community/blogs/aimsupport/entry/what_is_mqtt_and_how_does_it_work_with_websphere_mq?lang=en.

[18] "mosquitto - An Open Source MQTT v3.1 Broker," [Online]. Available: http://mosquitto.org/. [Acedido em 24 Agosto 2013].

[19] "mqtt - MQ Telemetry Transport - Documentação do mosquitto," [Online]. Available: http://mosquitto.org/man/mqtt-7.html. [Acedido em 18 Abril 2013].

[20] "databasedev.co.uk," [Online]. Available: http://www.databasedev.co.uk/image/northwind_relationships.gif. [Acedido em 13 Setembro 2013].

[21] B. Gentile, "Top 5 Myths About Big Data," 19 Junho 2012. [Online]. Available: http://mashable.com/2012/06/19/big-data-myths/. [Acedido em 3 Julho 2013].

[22] N. Hemsoth, "How Ford is Putting Hadoop Pedal to the Metal," 16 Março 2013. [Online]. Available: http://www.datanami.com/datanami/2013-03-16/how_ford_is_putting_hadoop_pedal_to_the_metal.html. [Acedido em 25 Agosto 2013].

[23] "Acquia: Examples of Big Data Projects," [Online]. Available: http://www.acquia.com/examples-big-data-projects. [Acedido em 25 Agosto 2013].

[24] "OneCare," [Online]. Available: http://www.onecare.pt/. [Acedido em 2013 Agosto 2013].

[25] "Toumaz SensiumVitals," [Online]. Available: http://www.toumaz.com/sensiumvitals%C2%AE-pilot-study#.Uh4qsNJwqSo. [Acedido em 29 Julho 2013].

[26] "Corventis AVIVO® Mobile Patient Management (MPM) System," [Online]. Available: http://www.corventis.com/us/avivo.asp. [Acedido em 11 Agosto 2013].

[27] "iRythm Zio Patch," [Online]. Available: http://www.irhythmtech.com/zio-solution/zio-patch/index.html. [Acedido em 20 Agosto 2013].

[28] "Plux BioSignals," [Online]. Available: http://www.biosignalsplux.com/. [Acedido em 4 Setembro 2013].

[29] I. Nonin Medical, "4100 Bluetooth Enabled Digital Pulse Oximeter OEM Specification and Technical Information," 2008. [Online]. Available: http://www.nonin.com/documents/4100%20Specifications.pdf. [Acedido em 10 Setembro 2013].

[30] "Java JDK," [Online]. Available: http://www.oracle.com/technetwork/java/javase/downloads/index.html. [Acedido em 13 Setembro 2013].

[31] "Oracle Mojarra JavaServer Faces," [Online]. Available: http://javaserverfaces.java.net/. [Acedido em 25 Agosto 2013].

[32] "ABI Research - More Than 30 Billion Devices Will Wirelessly Connect to the Internet of Everything in 2020," [Online]. Available: https://www.abiresearch.com/press/more-than-30-billion-devices-will-wirelessly-conne. [Acedido em 23 Agosto 2013].

[33] "MQTT.org," [Online]. Available: http://mqtt.org/. [Acedido em 24 Agosto 2013].

[34] "IBM - MQ Telemetry Transport (MQTT) V3.1 Protocol Specification," [Online]. Available: http://www.ibm.com/developerworks/webservices/library/ws-mqtt/index.html. [Acedido em 24 Agosto 2013].

[35] "Building Facebook Messenger," [Online]. Available: https://www.facebook.com/notes/facebook-engineering/building-facebook-messenger/10150259350998920. [Acedido em 25 Agosto 2013].

[36] "Wikipedia - MQ Telemetry Transport," [Online]. Available: http://en.wikipedia.org/wiki/MQ_Telemetry_Transport. [Acedido em 25 Agosto 2013].

[37] "Apache Active MQ," [Online]. Available: http://activemq.apache.org/. [Acedido em 24 Agosto 2013].

[38] "Secure Sockets Layer (SSL) - SearchSecurity - TechTarget," [Online]. Available: http://searchsecurity.techtarget.com/definition/Secure-Sockets-Layer-SSL. [Acedido em 3 Setembro 2013].

[39] "Xively," [Online]. Available: https://xively.com/. [Acedido em 25 Agosto 2013].

[40] "Hive MQ," [Online]. Available: http://www.hivemq.com/. [Acedido em 25 Agosto 2013].

[41] "IBM Integration Bus," [Online]. Available: http://www-03.ibm.com/software/products/us/en/integration-bus/. [Acedido em 25 Agosto 2013].

[42] "Wikipedia - IBM Lotus Expeditor," [Online]. Available: http://en.wikipedia.org/wiki/IBM_Lotus_Expeditor. [Acedido em 25 Agosto 2013].

[43] "Moquette," [Online]. Available: https://code.google.com/p/moquette-mqtt/. [Acedido em 25 Agosto 2013].

[44] "Rabbit MQ," [Online]. Available: http://www.rabbitmq.com/. [Acedido em 25 Agosto 2013].

[45] "Eclipse Paho," [Online]. Available: http://www.eclipse.org/paho. [Acedido em 25 Agosto 2013].

[46] "Fusesource MQTT Client," [Online]. Available: https://github.com/fusesource/mqtt-client. [Acedido em 25 Agosto 2013].

[47] "Google Gson," [Online]. Available: https://code.google.com/p/google-gson/. [Acedido em 30 Junho 2013].

[48] "Apache Tomcat," [Online]. Available: http://tomcat.apache.org/. [Acedido em 8 Setembro 2013].

[49] "opencsv," [Online]. Available: http://opencsv.sourceforge.net/. [Acedido em 25 Junho 2013].

[50] "Highcharts," [Online]. Available: http://www.highcharts.com/. [Acedido em 9 Setembro 2013].

[51] L. Costanzo, BRS Physiology, Wolters Kluwer Health/Lippincott Williams & Wilkins, 2010.

[52] "Wikipedia - Overhead (computação)," [Online]. Available: http://pt.wikipedia.org/wiki/Overhead_(computa%C3%A7%C3%A3o). [Acedido em 24 Agosto 2013].

[53] "TCP (Transmission Control Protocol) - SearchSecurity - TechTarget," [Online]. Available: http://searchnetworking.techtarget.com/definition/TCP.

Anexos

Anexo A

Resultados de Teste de Receção de Dados


```
Connected to tcp://84.9[redacted]:1883 with client ID gcl-subscriber
Subscribing to topic "isaMQTTUser/test/isa/#" qos 1
Press <Enter> to exit
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"0"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"1000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"2000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"3000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"4000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"5000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"6000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"7000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"8000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"9000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"10000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"11000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"12000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"13000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"14000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"15000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"16000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"17000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"18000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"19000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"20000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"21000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"22000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"23000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"24000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"25000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"26000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"27000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"28000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"29000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"30000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"31000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"32000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"33000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"34000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"35000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"36000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"37000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"38000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"39000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"40000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"41000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"42000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"43000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"44000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"45000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"46000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"47000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"48000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"49000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"50000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"51000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"52000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"53000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"54000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"55000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"56000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"57000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"58000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"59000"}
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378649810531","type":1,"aq_rate":1,"aq_stamp":"60000"}
```

Teste de Intervalo Temporal entre Envio e Entrega de Dados

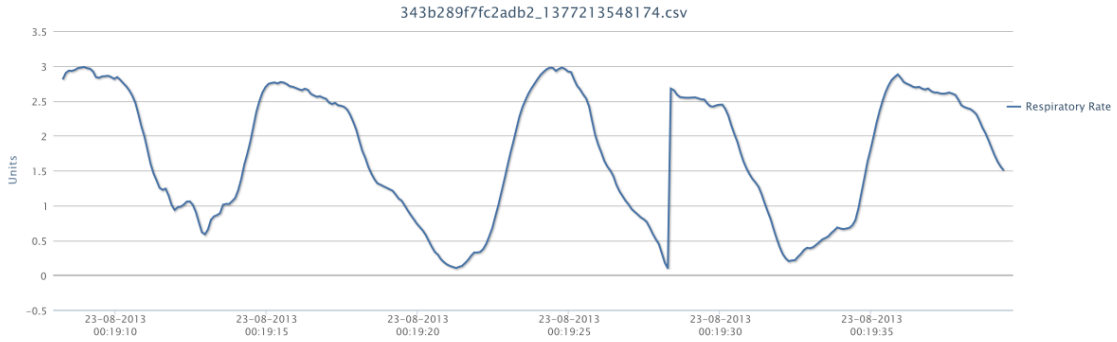
```
Connected to tcp://84.9[redacted]:1883 with client ID gcl-subscriber
Subscribing to topic "isaMQTTUser/test/isa/#" qos 1
Press <Enter> to exit
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649500440"} at 1378649500550
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649500548"} at 1378649500709
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649500709"} at 1378649500851
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649500851"} at 1378649500993
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649500993"} at 1378649501142
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649501142"} at 1378649501286
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649501292"} at 1378649501431
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649501435"} at 1378649501501
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649501501"} at 1378649501658
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649501658"} at 1378649501720
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649501719"} at 1378649501876
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649501876"} at 1378649502034
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649502034"} at 1378649502191
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649502191"} at 1378649502352
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649502354"} at 1378649502511
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649502511"} at 1378649502667
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649502668"} at 1378649502822
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649502822"} at 1378649502975
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649502975"} at 1378649503132
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649503132"} at 1378649503287
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649503287"} at 1378649503443
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649503443"} at 1378649503598
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649503599"} at 1378649503708
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649503709"} at 1378649503773
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649503774"} at 1378649503841
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649503841"} at 1378649503908
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649503909"} at 1378649503973
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649503973"} at 1378649504030
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649504030"} at 1378649504109
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649504110"} at 1378649504175
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649504176"} at 1378649504241
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649504241"} at 1378649504292
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649504293"} at 1378649504367
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649504367"} at 1378649504429
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649504430"} at 1378649504491
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649504554"} at 1378649504634
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649504635"} at 1378649504696
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649504696"} at 1378649504747
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649504749"} at 1378649504810
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649504810"} at 1378649504887
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649504886"} at 1378649504947
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649504947"} at 1378649505009
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649505007"} at 1378649505067
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649505066"} at 1378649505128
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649505129"} at 1378649505178
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649505178"} at 1378649505240
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649505240"} at 1378649505289
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649505289"} at 1378649505350
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649505350"} at 1378649505401
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649505400"} at 1378649505466
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649505464"} at 1378649505523
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649505523"} at 1378649505574
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649505573"} at 1378649505624
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649505624"} at 1378649505683
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649505684"} at 1378649505735
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649505735"} at 1378649505796
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649505795"} at 1378649505862
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649505861"} at 1378649505941
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649505941"} at 1378649506032
{"value":"97","imei":"658g334n1a8hhk8","in_stamp":"1378649500440","type":1,"aq_rate":1,"aq_stamp":"1378649506033"} at 1378649506135
```

Teste de Entrega de Dados após Re-conexão do Broker

```
Connected to tcp://84.91[redacted]5:1883 with client ID gcl-subscriber
Subscribing to topic "isaMQTTUser/test/isa/#" qos 1
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"0"} at 1378665420551
Press <Enter> to exit
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"1000"} at 1378665420941
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"2000"} at 1378665420951
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"3000"} at 1378665420954
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"4000"} at 1378665420956
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"5000"} at 1378665420959
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"6000"} at 1378665420961
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"7000"} at 1378665420964
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"8000"} at 1378665420966
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"9000"} at 1378665420970
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"10000"} at 1378665420973
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"11000"} at 1378665420975
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"12000"} at 1378665420978
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"13000"} at 1378665421032
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"14000"} at 1378665421035
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"15000"} at 1378665421038
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"16000"} at 1378665421040
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"17000"} at 1378665421043
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"18000"} at 1378665421047
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"19000"} at 1378665421052
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"20000"} at 1378665421085
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"21000"} at 1378665421087
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"22000"} at 1378665421131
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"23000"} at 1378665421134
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"24000"} at 1378665421136
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"25000"} at 1378665421139
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"26000"} at 1378665421141
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"27000"} at 1378665421144
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"28000"} at 1378665421147
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"29000"} at 1378665421150
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"30000"} at 1378665421153
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"31000"} at 1378665421155
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"32000"} at 1378665421158
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"33000"} at 1378665421211
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"34000"} at 1378665421296
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"35000"} at 1378665421300
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"36000"} at 1378665421304
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"37000"} at 1378665421306
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"38000"} at 1378665421309
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"39000"} at 1378665421312
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"40000"} at 1378665421315
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"41000"} at 1378665421319
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"42000"} at 1378665421322
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"43000"} at 1378665421324
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"44000"} at 1378665421327
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"45000"} at 1378665421329
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"46000"} at 1378665421332
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"47000"} at 1378665421335
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"48000"} at 1378665421338
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"49000"} at 1378665421341
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"50000"} at 1378665421343
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"51000"} at 1378665421346
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"52000"} at 1378665421354
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"53000"} at 1378665421367
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"54000"} at 1378665421372
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"55000"} at 1378665421627
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"56000"} at 1378665421631
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"57000"} at 1378665421635
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"58000"} at 1378665421638
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"59000"} at 1378665421642
{"value":"98","imei":"658g334n1a8hhk8","in_stamp":"1378662548731","type":1,"aq_rate":1,"aq_stamp":"60000"} at 1378665421645
```

Anexo B

Exemplo de gráfico de Taxa Respiratória



O que se representa é a variação de volume respiratório (valores altos correspondem a inspirações, e valores baixos a expirações).

Anexo C

Parte de Ficheiro do tipo CSV de Oximetria de Pulso

658g334n1a8hhk8_1378649810532.csv

```
"IMEI","658g334n1a8hhk8","",""
"INITIAL TIME","1378649810532","",""
"TYPE","AQUISITION TIME GAP","AQUISITION RATE","VALUE"
"1","0","1","98"
"1","1000","1","97"
"1","2000","1","98"
"1","3000","1","98"
"1","4000","1","98"
"1","5000","1","99"
"1","6000","1","98"
"1","7000","1","98"
"1","8000","1","98"
"1","9000","1","98"
"1","10000","1","98"
"1","11000","1","97"
"1","12000","1","98"
"1","13000","1","98"
"1","14000","1","99"
"1","15000","1","98"
"1","16000","1","98"
"1","17000","1","98"
"1","18000","1","98"
"1","19000","1","98"
"1","20000","1","98"
"1","21000","1","98"
"1","22000","1","98"
"1","23000","1","98"
"1","24000","1","98"
"1","25000","1","98"
"1","26000","1","98"
"1","27000","1","98"
"1","28000","1","98"
"1","29000","1","98"
"1","30000","1","98"
"1","31000","1","98"
```

Parte de Ficheiro do tipo CSV de Oximetria de Pulso e Frequência Cardíaca

343b289f7fc2adb2_1377037248381.csv

```
"IMEI","343b289f7fc2adb2","",""
"INITIAL TIME","1377037248381","",""
"TYPE","AQUISITION TIME GAP","AQUISITION RATE","VALUE"
"1","0","1","97"
"2","0","1","61"
"1","1000","1","97"
"2","1000","1","61"
"1","2000","1","97"
"2","2000","1","61"
"1","3000","1","97"
"2","3000","1","61"
"1","4000","1","97"
"2","4000","1","61"
"1","5000","1","97"
"2","5000","1","61"
"1","6000","1","97"
"2","6000","1","62"
"1","7000","1","97"
"2","7000","1","63"
"1","8000","1","97"
"2","8000","1","64"
"1","9000","1","97"
"2","9000","1","64"
"1","10000","1","97"
"2","10000","1","65"
"1","11000","1","97"
"2","11000","1","65"
"1","12000","1","97"
"2","12000","1","63"
"1","13000","1","97"
"2","13000","1","61"
"1","14000","1","97"
"2","14000","1","59"
"1","15000","1","97"
"2","15000","1","58"
```

Parte de Ficheiro do tipo CSV de Eletrocardiograma

343b289f7fc2adb2_1377214796906.csv

```
"IMEI","343b289f7fc2adb2"
"INITIAL TIME","1377214796906"
"TYPE","AQUISITION TIME GAP","AQUISITION RATE","VALUE"
"4","10","100","1.527099609375"
"4","20","100","1.575439453125"
"4","30","100","1.61279296875"
"4","40","100","1.648681640625"
"4","50","100","1.647216796875"
"4","60","100","1.62451171875"
"4","70","100","1.57177734375"
"4","80","100","1.527099609375"
"4","90","100","1.477294921875"
"4","100","100","1.453125"
"4","110","100","1.43115234375"
"4","120","100","1.43115234375"
"4","130","100","1.42529296875"
"4","140","100","1.4326171875"
"4","150","100","1.427490234375"
"4","160","100","1.436279296875"
"4","170","100","1.434814453125"
"4","180","100","1.443603515625"
"4","190","100","1.44140625"
"4","200","100","1.4501953125"
"4","210","100","1.447265625"
"4","220","100","1.4560546875"
"4","230","100","1.455322265625"
"4","240","100","1.465576171875"
"4","250","100","1.462646484375"
"4","260","100","1.474365234375"
"4","270","100","1.472900390625"
"4","280","100","1.484619140625"
"4","290","100","1.482421875"
"4","300","100","1.4912109375"
"4","310","100","1.48974609375"
"4","320","100","1.49853515625"
```

(...)

343b289f7fc2adb2_1377214796906.csv

```
"4","7120","100","1.53662109375"
"4","7130","100","1.517578125"
"4","7140","100","1.529296875"
"4","7150","100","1.510986328125"
"4","7160","100","1.52783203125"
"4","7170","100","1.510986328125"
"4","7180","100","1.52490234375"
"4","7190","100","1.502197265625"
"4","7200","100","1.395263671875"
"4","7210","100","1.251708984375"
"4","7220","100","1.647216796875"
"4","7230","100","2.93115234375"
"4","7240","100","2.952392578125"
"4","7250","100","2.642578125"
"4","7260","100","1.69482421875"
"4","7270","100","1.40771484375"
"4","7280","100","1.257568359375"
"4","7290","100","1.177734375"
"4","7300","100","1.182861328125"
"4","7310","100","1.172607421875"
"4","7320","100","1.201171875"
"4","7330","100","1.201904296875"
"4","7340","100","1.232666015625"
"4","7350","100","1.237060546875"
"4","7360","100","1.267822265625"
"4","7370","100","1.269287109375"
"4","7380","100","1.302978515625"
"4","7390","100","1.306640625"
"4","7400","100","1.34033203125"
"4","7410","100","1.34326171875"
"4","7420","100","1.3798828125"
"4","7430","100","1.386474609375"
"4","7440","100","1.427490234375"
"4","7450","100","1.50732421875"
"4","7460","100","1.546875"
```

Parte de Ficheiro do tipo CSV de Taxa Respiratória

343b289f7fc2adb2_1377213143566.csv

```
IMEI,343b289f7fc2adb2,,
INITIAL TIME,1377213143566,,
TYPE,AQUISITION TIME GAP,AQUISITION RATE,VALUE
6,100,10,1.939453125
6,200,10,1.908691406
6,300,10,1.897705078
6,400,10,1.830322266
6,500,10,1.689697266
6,600,10,1.569580078
6,700,10,1.509521484
6,800,10,1.482421875
6,900,10,1.471435547
6,1000,10,1.398925781
6,1100,10,1.271484375
6,1200,10,1.190917969
6,1300,10,1.104492188
6,1400,10,1.078857422
6,1500,10,1.169677734
6,1600,10,1.458251953
6,1700,10,1.785644531
6,1800,10,2.1328125
6,1900,10,2.481445313
6,2000,10,2.721679688
6,2100,10,2.850585938
6,2200,10,2.920166016
6,2300,10,2.956054688
6,2400,10,2.975830078
6,2500,10,2.985351563
6,2600,10,2.982421875
6,2700,10,2.974365234
6,2800,10,2.986083984
6,2900,10,2.967773438
6,3000,10,2.983154297
6,3100,10,2.986083984
```

Anexo D

ELSEVIER

Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Technology 00 (2013) 000–000

Procedia Technology

www.elsevier.com/locate/procedia

CENTERIS 2013 - Conference on ENTERprise Information Systems / HCIST 2013 - International Conference on Health and Social Care Information Systems and Technologies

System of acquisition, transmission, storage and visualization of Pulse Oximeter and ECG data using *Android* and MQTT

Daniel Barata[a], Gonçalo Louzada[a], Andreia Carreiro[b,c*], António Damasceno[b]

[a]*University of Coimbra, Coimbra, Portugal*
[b]*ISA Intelligent Sensing Anywhere, S.A, Coimbra, Portugal*
[c]*Institute for Systems Engineering and Computers(INESCC), Coimbra, Portugal*

Abstract

The prevention of medical disorders significantly increases both life quality and expectancy. Continuous monitoring of biomedical parameters and/or vital signs of a patient, and easy access to medical history is thus of major importance, whilst enabling a person's normal routine. The development of remote healthcare sensors used to monitor specific conditions and with digital interface for prompt data access, arises as a valuable asset to respond to such requirements. In this paper we present a system of health data collection, transmission and storage designed for electrocardiography (ECG) and Pulse Oximetry results.
Our goal is to address this challenge by creating a system for acquiring and transmitting data via Bluetooth to an *Android* mobile platform, which sends it to a remote server, where the data is stored in a database, and becomes available for visualization This enables any remote user to access vital data on a patient without being physically present.

Keywords: ECG; Pulse Oximetry; Bluetooth; *Android*; Internet; MQTT; Java

1. Introduction and Objectives

The high rate of evolution of health-related areas has led to increased quality and life expectancy [1]. Prevention of diseases and complications assumes an increasingly important role in order to keep these values high enough to satisfy society needs. The possibility of using portable medical devices that let a person do its routine while monitoring health parameters is a strong advantage in prevention role. This supports the idea that there is a lot of potential in developing medical devices for remote monitoring.

The objective of this work is to develop an effective mechanism for data transmission between a monitoring device, which does the acquisition of health parameters, and a remote receiver, which allows storage and access to the acquired data, for instance, by a doctor geographically distant from the patient. To achieve this, an already existent product, OneCare, will serve as base to all the work developed.

The paper is structured in 5 different sections: introduction section; in section 2 we give an overview of the state of the art regarding the technology used in the project; in section 3 we contextualize the line of products in which our project is inserted, explaining why our project adds value to the existent solution; in section 4 we explain our solution in detail; finally in section 5 we describe the conclusions and the future work.

2. State of the Art

2.1. Android Mobile Platforms

This application is targeted to smartphones and *tablet*s that run the operating system *Android*. These devices are becoming increasingly useful and are a symbol of technological progress. *Android* is one of the operating systems used in *tablet* and smartphones, and now is in possession of the multinational company, Google. There are other operating systems such as iOS, kindle, etc. However, only *Android* offers the following advantages:

- Allows its use in various devices, because it is open-source;
- Is in this mainstream technology branch, holding about 46.9% of users [2];
- The development of applications for this operating system is much easier;
- Evolution and significant growth, to an increasingly broad range of electronic devices.

2.2. Bluetooth

Bluetooth is an application that, by using short wavelength radio transmissions, allows mobile devices to exchange data wirelessly over short distances, and since 1994, the date of its creation, it began to be used as a main feature of mobile phones [3]. Resuming, Bluetooth is a wireless, low power, low cost technology for exchange of information over short distances and, is the perfect technology to do the data transmission from the medical devices to the smartphones/*tablet*s.

2.3. Android in Healthcare

Android is already being used in healthcare solutions, and current scientific articles regarding this subject reflect that fact. There are some developments with focus on acquisition and visualization of health parameters on smartphones and PDA's, such as the ones presented by Lo et al. [4] and Kai et al. [5]. At a commercial level, there are *Android* applications for various purposes and the healthcare is clearly an area with plenty of incidence. However, the main focus is not telemetry systems and the applications are mainly just informative or a guide for the user in question related to healthcare.

Something that is more similar to our concept is an application called Mobile ECG Telemetry Solution (METS), developed by MegaKoto, which aims to collect various parameters, including ECG and Oximetry, and sending the user to a database where monitoring is continuous [6].

So, join medical devices that have Bluetooth technology, *Android* and a transmission technology as MQTT, which has all the advantages already mentioned, has everything to be an innovative and effective product in their market.

2.4. MQTT

MQTT is a machine-to-machine (M2M)/"Internet of Things" connectivity protocol. It was designed as an extremely lightweight publish/subscribe messaging transport. It is useful for connections with remote locations where a small code footprint is required and/or network bandwidth is at a premium. Developed by Dr. Andy Stanford-Clark of IBM, and Arlen Nipper of Arcom (now Eurotech), in 1999 [7], MQTT's main functional principle is the existence of topics in an intermediate entity named *broker*. Basically, when a client subscribes or publishes on a certain topic, that topic is registered on the *broker*. Multiple subscriptions can be made by the same client to different topics, as well as different clients can subscribe the same topic. This works the same way to publishers. So the clients subscribing the topic X will receive every message published under that topic [8]. Concluding, the *broker* and MQTT act as a simple, common interface for everything to connect to, as said in the website of one of the available *broker*s, *Mosquitto* [9].

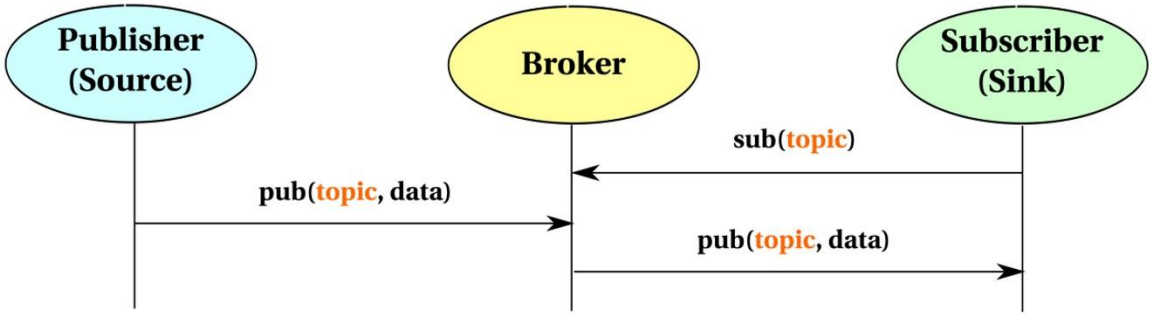


Fig. 1: Topic-based Pub/Sub Communication Model [8].

2.5. MQTT in Healthcare

Currently there are few reports of MQTT usage in healthcare systems. The search for articles regarding this subject has revealed itself unsuccessful, but a wider search helped us to find a known case of MQTT being a tool in healthcare systems. IBM developed a system that allows the patients of the St. Jude Medical Center carrying pacemakers or cardioverter defibrillators, to be monitored continuously from their homes. To achieve this, a special device - Merlin™@home - is attached to the referred healthcare devices and sends a signal and respective data when any unusual event happens directly to the medical center [10]. This allows not only higher level of patient care and early diagnosis of problems, but also improved administrative efficiency and maintenance and help in conforming to standards, with easy integration of data [11]. At its Software section, the MQTT official web page reports 100,000 heart pacemakers monitored via MQTT at St. Jude [7].

There were not found more concrete, well documented applications of MQTT in health, making our project innovative and challenging, although its structure can be compared to the one described in IBM's RedBook Building Smarter Planet Solutions with MQTT and IBM WebSphere MQ Telemetry [12].

3. OneCare and General Architecture

OneCare is a line of products specialized in transmitting health parameters remotely. The OneCare line of products consists of two main products: OneCare-Sensing and OneCare-Safe. These two products belong to the concept of the tele-assistance service, conceived to improve the quality of life, health and security of its users with verified reliability [13].

3.1. OneCare-Sensing

OneCare-Sensing enables continuous monitoring of the user's health status, allowing the premature detection of potential risk situations. The data can be accessed remotely through a web portal at any time, by entering a username and password. In this way, there are better chances of an effective and on-time intervention to avoid dangerous health situations for the end-user.

Every monitoring frequency is adjusted to each case, depending on the type of user. The health parameters are obtained through different wireless medical devices: a blood-pressure monitor, a weighing machine or any other device for biosignal acquisition that supports Bluetooth. Additionally, an *Android tablet* can be used for visualization, where the user can see relevant gathered information, checking if measurements were well done and sent to the database.

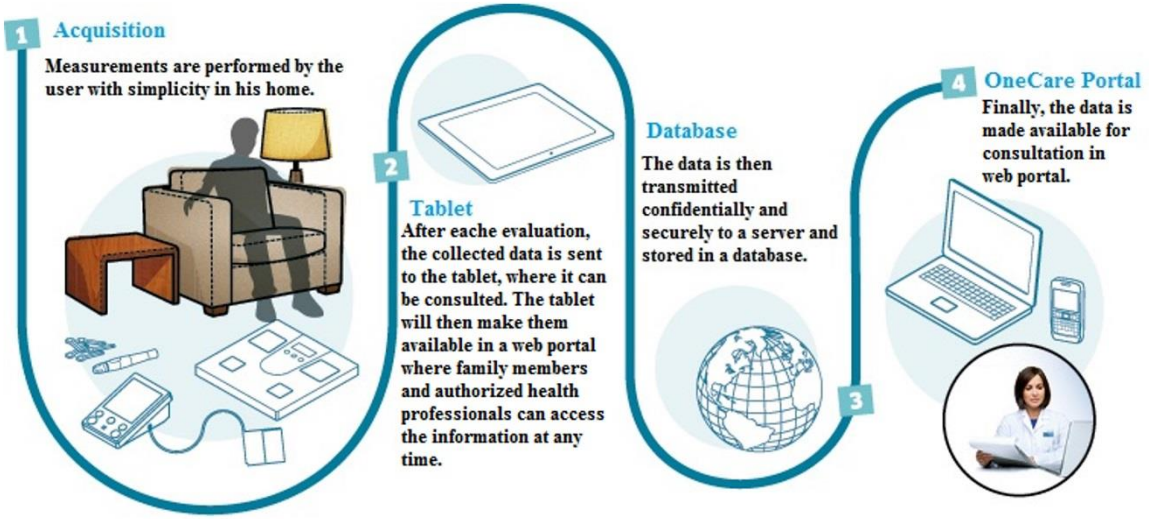


Fig. 2: OneCare-Sensing schematic [13].

An important functionality is the system's capacity of alert emission when the standard behavior of the measured parameters shows any significant deviation. These alerts are registered in a database and, besides becoming available in the web portal, can be sent over SMS or email to a caretaker, family member or any other desired individual. In this way, the user can be immediately contacted in order to avoid false alarms, establish a relation between symptoms and the collected data, or confirm the need for immediate help. The alert conditions are previously established by the caretakers and may include various levels for each user. For having an automatic and customized alarm program, the system doesn't require daily validation of all collected data by the caretaker. Instead, there is only need to check the alerts sent by the system, which have different severity grades. In this way, OneCare-Sensing provides a way of increasing monitoring of risks, without increasing the costs of needing more healthcare personnel necessary.

3.2. OneCare-Safe

OneCare-Sensing is an easy-usage portable device that allows a caretaker to follow the well-being of its user. It is specially indicated to elders or people with special needs. This device possesses an alert button (which activates an aid request, either at home or outside), a fall detector, a GPS tracking device (activated with the alert button) and the ability to send and receive text messages and voice calls to or from predetermined contacts (caretakers, family or any other individual). With these functions, this device allows constant follow of the user, whether in or outside the user's house, as long as the user carries it.

4. Added value

Based in the OneCare concept, there are improvements that can be made. The goal of our project is to improve the product by adding electrocardiography (ECG) and Pulse Oximeter modules to the product, renew the way of sending the data to the central database and the way it is stored as well, thus making it more complete and effective. In order to do this, a new architecture will be established: an *Android* application that receives the health parameters data from portable medical devices using Bluetooth. This application also has the functionality to show the values of greater interest to the user, while sending them in real time to the product's server by using the Message Queuing Telemetry Transport (MQTT) protocol. In that server there will be a client responsible for receiving and storing the data appropriately. So we can distinguish two parts: the acquisition and data-sending module, and the reception, storage and visualization

module. All this will be integrated within the existent OneCare platform, making use of its already developed resources and web portal.

Fig. 3. Schematic representation of the project.

4.1. Portable Medical Devices

The first stage of the system is to acquire the vital signals ECG and Pulse Oximetry. Pulse Oximetry quantifies the saturation of hemoglobin in circulation, meaning, measures (in percentage) the quantity of oxygen present in arterial blood [14]. ECG measures the electrical activity of the heart, allows the deduction of an individual's heart rate and the by observation of the recorded data permits the evaluation their regularity or the detection of many cardiac arrhythmias [15].

To obtain these two measures, specific portable medical devices with a Bluetooth sender module are needed. Nonin 4100 is used as a Pulse Oximeter and a Plux device as an Electrocardiograph.

4.2. Android Application, Bluetooth receiver and MQTT Publisher

This application is targeted to smartphones and *tablet*s that run the operating system *Android*. The *Android* application was programmed using *Android* Development Tools (ADT), a plugin for Eclipse, in order to use the Java programming language. Our application has two distinct functions, one is to receive data from the medical devices already stated above, and the other is to publish the data using MQTT technology.

After the application initialization there is a button to start the service in order to verify if the MQTT connection was established and, in case of success, the acquisition of health parameters can be initialized. With that, a part of the program creates a RFCOMM Bluetooth socket that starts the data acquisition and then, at same time:

- Data is collected into an array of objects. A specific object Class for the packets (considered each point of the future graphic) is created, containing all the information and to excluding unnecessary values;
- A packet is stored, the essential values and all needed timestamps are gathered in a *JSON* object, transformed in a *JSON*String and then sent to an MQTT Publisher Class.
- The values of interest for the patient are sent by a handler to the Main Activity, in order to show the measurement in a graphic interface.

In MQTT Publisher, for each value received, a MQTT client is created with the options specified for each *broker*. Then the *JSON*String received is then published in a topic (we use a random topic but in future a topic can be created for each client). This process is repeated every time a *JSON*String is received.

4.3. Broker, Server and Java Client

The *broker* installed in our server is *Mosquitto*. In addition, it has a client running, responsible for subscribing the desired topics, receiving the data, decoding it (since it will be sent as a *JSON* string) and storing it appropriately. The client's programming language is Java, because of the functionalities it allows and so it is compatible with the already existent structures.

First, we performed standard tests to understand the MQTT principles with online test *broker*s (test.*Mosquitto*.org, *broker*.mqttdashboard.com) and the command line: subscribing to topics and publishing messages in those topics. After that, research was done to evaluate what was already made regarding Java MQTT clients in order to define a starting point. We came across different software (the Software section of the MQTT official website has some options, but we searched the web as well), such as Java client Application Programming Interfaces (API's) and *broker*s, which are listed in tables 1.

To comply with the standards of the company we are developing for, we were instructed to work with the *Mosquitto broker* (though initially the company had Apache MQ running, later that was changed). To select the Java Client API, our criteria was to choose one actual, recent, *not dead* (meaning, it is used and under constant evolution) and that had all the functionalities needed. For that purpose, the one we found

more suitable was the Eclipse Paho API, which is Open-Source.

Table 1. List of MQTT *Broker*s and Java Client API's

*Broker*s	Type of License	Java Client API's
Active MQ	Open-Source	Eclipse Paho - a Java client developed by the Eclipse Foundation
Xively (formerly Cosm)	Commercial	MeQanTT
Hive MQ	Commercial	mqtt-client - a Fusesource Java MQTT client with a variety of API styles
IBM Lotus Expeditor micro *broker*	Commercial	moquette-mqtt - a Java cliente
IBM Websphere Message *Broker*	Commercial	IBM WebSphere MQ Telemetry provides a Java client API
Moquette	Open-Source	IBM Lotus Expeditor micro *broker* includes two Java client APIs
Mosquitto	Open-Source	
Rabbit MQ	Open-Source	
RSMB - Really Small Message *Broker*	Commercial	

Chosen the *broker* and the API, it was time to start developing the code itself. After more research, the code started to acquire shape, first by simply subscribing and publishing in topics, and receiving messages. Then, using the properties of the Paho API, the code was set to login into the company's servers (internal and external), which require a username and password. The client was then set to have persistent subscriptions, so it would not lose data sent while the client was offline. The client is already prepared to authenticate via SSL, although that part of the code is currently inactive since in the moment communication is made via TCP. In the future, this will be changed to meet the security requirements of medical data transmission. After establishing communication with the company servers, the next step was preparing the code to receive the messages sent as *JSON* Strings. That was achieved using an external API named gson-2.2.2 (https://code.google.com/p/google-gson/) and establishing a consensus over the formatting of the data sent from the sources. This includes the id of the device used, the timestamp of the acquisition and the desired health values. The timestamp is particularly important to assure that the data is correct and hasn't suffered any deformation over the transmission (such as duplication or time misplacement). After this, the final step was making the client write the data to a file for each source. This was done using the *FileWriter* package.

Between all the phases and in the end, consistency tests we're performed to ensure everything was working as predicted. Some of these tests consisted of sending data from 20 different virtual sources at the same time to ensure that the client received and registered every piece of data, or sending data while the client was offline and then restarting the client to see if it received the data sent while it was down. Until now, the client passed all the tests.

5. Conclusion and Future Work

After performing tests, the results are promising. Although we only used the Pulse Oximeter, the data was collected by the *Android tablet* and immediately processed and sent. The delivery time was very short (about a second in most cases) and this was proved by subtracting the time acquisition timestamp to the delivery timestamp. No data has been lost over the MQTT transmission.

Now that the ECG material finally arrived, we will soon begin to collect ECG data and adapting the project to the needs of this different kind of data. The storage format is under revision, but almost complete. Simultaneously the visualization tool is already being developed and will be integrated with the OneCare product portal soon. Besides these tasks, the communication methods will be reviewed to ensure no data is lost, and both the *Android* application and the Java Client will be evaluated to determine if improvements can be made.

Acknowledgements

We would like to thank ISA for providing a good environment and the facilities to complete this project.

References

[1] "Pordata" Available: http://www.pordata.pt/Portugal/Esperanca+de+vida+a+nascenca+total+e+por+sexo-418. Accessed 01/05/2013.

[2] "Go Gulf Blog" Available: http://www.go-gulf.com/blog/smartphone/ Accessed 06/05/2013.

[3] "Mobile Innovation" Available: http://www.themobileinnovation.net/importance-of-bluetooth-technologies-on-mobile-phones. Accessed 06/05/2013.
[4] Lo BPL, Thiemjarus S, King R, Yang GZ. Body Sensor Network. A Wireless Sensor Platform for Pervasive Healthcare Monitoring.
[5] Kai L, Xu Z, Yuan W, Suibiao H, Ning G, Wangyong P, Bin L, Hongda C. A System of Portable ECG Monitoring Based on Bluetooth Mobile Phone. IT in Medicine and Education 2011; volume 2; p.309 – 312.
[6] "MegaKoto" Available: http://www.megakoto.fi/en/page/1747. Accessed 06/07/2013.
[7] "MQTT Web Page". Available: http://mqtt.org/. Accessed 06/05/2013.
[8] Hunkeler, U, Hong Linh Truong, Stanford-Clark, A. MQTT-S – A Publish/Subscribe Protocol For Wireless Sensor Networks. 2008 3rd International Conference on Communication Systems Software and Middleware and Workshops, Jan. 2008, pp.791-798.
[9] "*Mosquitto* Web Page". Online: http://*Mosquitto*.org/ Accessed 06/05/2013.
[10] "Readwrite: MQTT Poised For Big Growth - an RSS For Internet of Things?" Available: http://readwrite.com/2009/07/22/mqtt_poised_for_big_growth#awesm=~oaLUs1UniPCQ5d. Accessed: 06/07/2013.
[11] "Wiki Eclipse" Available: http://wiki.eclipse.org/images/5/53/M2M_-_MQTT_Analyst_Briefing_Package.pdf. Accessed: 06/07/2013.
[12] Lampkin, Valerie; Leong, Weng Tat; Olivera, Olivera; Rawat, Sweta; Subrahmanyam, Nagesh; Xiang, Rong. Building Smarter Planet Solutions with MQTT and IBM WebSphere MQ Telemetry, 1st edition, p 14-15
[13] "OneCare Web Page" Available: http://www.onecare.pt/pt. Accessed 01/05/2013.
[14] "Pulse Oximetry Wikipedia Page" Available: http://en.wikipedia.org/wiki/Pulse_oximetry. Accessed 02/05/2013.
[15] "Electrocardiography Wikipedia Page" Available: http://en.wikipedia.org/wiki/Electrocardiography. Accessed 02/05/2013.