FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Aplicações Bluetooth e Perspectivas de Evolução: Um Novo Serviço de Emergência em Plataformas Open-Source

Adelino António Vale Vieira do Couto
Licenciado em Engenharia Electrónica Industrial pela
Universidade do Minho

Dissertação submetida para satisfação parcial dos
requisitos do grau de mestre em
Redes e Serviços de Comunicação

Dissertação realizada sob a supervisão do
Professor Doutor Nuno Teixeira de Almeida,
do Departamento de Engenharia Electrotécnica e de Computadores
da Faculdade de Engenharia da Universidade do Porto

Porto, Fevereiro de 2008

À minha mulher e filho…

Aos meus pais e irmãs…

RESUMO

Desde o início da História da Humanidade que o Homem tem uma capacidade inata para inventar e inovar. Apesar de longínquas, descobertas como a roda, o fogo e a lança, estão ainda hoje presentes na vida quotidiana. Obviamente que o trabalho descrito nesta dissertação não tem como objectivo igualar qualquer uma daquelas, contudo, é uma ideia nova que poderá ser utilizada pela sociedade civil, trazendo valor acrescentado para os seus serviços de emergência.

O trabalho consiste no desenvolvimento de uma nova aplicação que servirá como auxílio às funções e actividades dos serviços de emergência médica de cada país. Esta aplicação consiste, basicamente, no envio de sinais Bluetooth, por parte de um veículo sinistrado, sendo aqueles sinais recebidos nos veículos de emergência.

O trabalho desenvolvido começa por apresentar uma visão da utilidade e do impacto que a aplicação poderá ter nos serviços de emergência e faz, também, o seu enquadramento, apontando as tecnologias existentes que poderão ser utilizadas para a sua implementação.

O desenvolvimento da aplicação é apresentado, sendo referida a metodologia de construção da mesma, que assentou na criação de um sistema cliente-servidor. Do lado cliente, existe uma aplicação de inserção, armazenamento e transmissão de informação, e do lado servidor, existe uma aplicação que cria o serviço prestado ao cliente, recebendo e processando aquela informação.

É demonstrada a implementação do serviço sendo utilizadas como ferramentas para a construção do mesmo as plataformas *open-source*: Linux, e Python integrada com o módulo *Bluetooth* e *Imaging*.

Foram realizados testes ao serviço desenvolvido revelando-se uma capacidade de comunicação eficiente, com um alcance máximo de 250 metros entre o local onde os veículos de emergência recebem o sinal Bluetooth, e o local do acidente. Os tempos totais de envio, recepção e visualização dos dados variam entre os 3,99s até aos 17,39s, nos testes em laboratório, e dos 6,29s até aos 18,01s, nos testes de simulação de um cenário real.

A dissertação termina com a apresentação das conclusões e ideias para trabalhos no futuro, os quais poderão melhorar a aplicação desenvolvida e dotá-la de novas funções.

ABSTRACT

Since the beginning of the History of the Mankind, the Man has an innate capacity to innovate and to invent. Although far away, discoveries such as the wheel, the fire and the spear, are still today present in our quotidian life. Obviously, the work that follows does not have the objective to equal any one of them. However is a new idea that could be used by the Civil Society, bringing value to the emergency services.

This work consists in the development of a new application that will serve as aid to the functions and activities of the services of medical emergency of each country. Basically, this application, consists on the sending of Bluetooth signals by the injured vehicle, being this signals, received by the emergency vehicle.

The developed work starts presenting a vision of the utility and the impact that the application shall have in the Society and makes its framing, pointing the existing technologies that could be used for its implementation.

The development of the application, is presented, being related the methodology of construction of the same one, that is based in a client-server system. Where on the side of the client, exists the application of insertion, storage and information transmission, and on the server side exists the application that creates the service given to the client, receiving and processing that information.

Is demonstrated the implementation of the system, being used as tools for the construction of the same, the *open-source* platforms: Linux, and Python integrated with the module *Bluetooth* and *Imaging*.

The tests to the system had disclosed a capacity of efficient communication having the system a maximum reach of 250 meters since the place where the emergency vehicles receive the Bluetooth signal, until the local of the accident. The periods of sending and reception of data vary between 3,99s until 17,39s in the tests of laboratory and 6,29s until 18,01s in the tests of simulation of the real scene.

The dissertation finishes with the presentation of the conclusions and ideas for work in the future that can improve and endow, of new functions, the developed application.

AGRADECIMENTOS

A elaboração desta dissertação não seria possível sem a colaboração directa ou indirecta de algumas pessoas, por isso, quero aqui deixar a minha palavra de gratidão para elas.

Quero começar por agradecer ao Professor Nuno Almeida, que no papel de orientador foi o meu principal parceiro.

Em segundo lugar quero expressar o meu agradecimento a Albert S. Huang, investigador no MIT (Massachusets Institute of Technology) desde 2003, que desenvolveu o módulo Bluetooth para a linguagem de programação Phyton, e que influenciou a construção da aplicação desenvolvida, na área das comunicações.

Em terceiro lugar agradeço aos meus amigos Ricardo Cruz, Luís Maia, Luís Monteiro, Raquel Faria e Manuel Silva pela cedência das suas fotografias para apresentação durante a dissertação.

Por último deixo o agradecimento à minha família que me proporcionou as melhores condições para executar o trabalho.

Adelino António Vale Vieira do Couto

ÍNDICE

ÍNDICE DE FIGURAS

LISTA DE ABREVIATURAS

ANACOM	Autoridade Nacional de Comunicações
AVC	Acidente Vascular Cerebral
FHSS	Frequency Hopping Spread Spectrum
GNR	Guarda Nacional Republicana
GSM	Global System for Mobile Communications
INEM	Instituto Nacional de Emergência Médica
JPEG	Joint Photographic Experts Group
PSP	Polícia de Segurança Pública
RFCOMM	Radio Frequency Communication
SEB	Serviço de Emergência Bluetooth
SDP	Service Discovery Protocol
SIG	Bluetooth Special Interest Group
SIRESP	Sistema Integrado das Redes de Emergência e Segurança de Portugal
UMTS	Universal Mobile Telecommunications System
USB	Universal Serial Bus
UUID	Universally Unique Identifier

Capítulo 1

Introdução

1 INTRODUÇÃO

A elaboração desta tese de mestrado teve como principal objectivo a criação e implementação de um novo serviço baseado na tecnologia de comunicação rádio Bluetooth**,** serviço esse que deverá ser implementado sobre plataformas *open-source*.

Além das vantagens convencionais, como a substituição de cabos, o Bluetooth também permite novas formas de troca ou de acesso à informação. Efectivamente, o atributo de mobilidade de quase todos os dispositivos equipados com Bluetooth, pode potenciar, em função do espaço fisíco ocupado num determinado instante, uma interacção mais dinâmica destes, com os sistemas (ou equipamentos) em posição estática. Um bom exemplo, desta interacção, será um equipamento Bluetooth trocar informação contextual de forma automática com outro dispositivo Bluetooth que se aproxime dele.

Existem imensas áreas onde esta tecnologia pode ser usada. Por isso, e depois de uma ponderação cuidadosa, acabou por ser escolhida a área da prestação de serviços de emergência médica a veículos sinistrados, área esta que é cada vez mais vital para a Sociedade Humana e donde se pode retirar enormes benefícios com o uso desta tecnologia.

O objectivo passa por desenvolver um sistema Bluetooth composto por duas partes que interagem de forma automática entre si, nomeadamente os serviços de emergência e as pessoas que precisam dos seus serviços. Isso será possível, dotando os veículos de emergência e os veículos civis, do referido sistema, ao qual decidimos chamar *SEB* (*Serviço de Emergência Bluetooth*).

Neste sistema, os veículos civis assumem a parte estática, pois apenas vão ter um papel activo no momento em que existe uma colisão, que danifique seriamente os veículos. Os veículos de emergência são a parte móvel, pois irão interagir com a parte estática no momento em que se dirigem aos veículos sinistrados para socorrer os seus ocupantes.

Esta interacção vai permitir a troca de informação contextual entre as duas partes, relativa aos ocupantes dos veículos sinistrados, fazendo-a chegar aos profissionais dos serviços de emergência, antes de estes chegarem ao local do sinistro.

Não é intenção deste documento explicar em pormenor a tecnologia Bluetooth [10], mas elucidar o seu leitor para os objectivos e para a forma como este trabalho foi desenvolvido e implementado.

De seguida, apresenta-se breve informação sobre a estrutura e capítulos da dissertação.

Capítulo 1: **Introdução**: apresentação do projecto, descrição abstracta do funcionamento do serviço e descrição genérica da estrutura da dissertação.

Capítulo 2: **Caracterização do Problema e seu Contexto**: enfoque sobre a existência de um problema, contextualização, apresentação e discussão de soluções para o mesmo.

Capítulo 3: **Arquitectura do Sistema e Abordagem Técnica**: descrição da composição do sistema, os seus pilares e elementos técnicos essenciais.

Capítulo 4: **Desenvolvimento e Implementação do Sistema**: este é o capítulo da construção do sistema, sendo mostrado como todos os blocos que o compõem são ligados entre si. Explica também a forma de implementação de cada um deles.

Capítulo 5: **Validação e Testes ao Sistema**: descrição pormenorizada dos cenários de testes e demonstração do sistema. Apresentação de resultados.

Capítulo 6: **Conclusões:** apresentação das conclusões de todo o trabalho, assim como das ideias para o futuro, de modo a poder melhorar o trabalho desenvolvido.

Capítulo 2

Caracterização do Problema e seu Contexto

2.1 Tecnologias

2.2 Linux - Kurumin **7**

2.3 Cenários de Actuação

2.4 Entidades que Podem Usufruir do SEB

2 CARACTERIZAÇÃO DO PROBLEMA E SEU CONTEXTO

Hoje em dia, na sociedade em que vivemos, a informação é algo que pode ter um valor incalculável. Muitas vezes, informação significa também poder. Poder de decisão, entre a escolha certa e a escolha errada.

Nunca, como agora, o mundo esteve tão preenchido pelas tecnologias de comunicação. Elas rodeiam-nos, para onde quer que se vá, contudo, o seu nível de aproveitamento ainda é muito reduzido, nomeadamente no que diz respeito a questões sociais e humanas.

Existem inúmeras actividades que podem ser melhoradas, desde que surja uma ideia nova que interligue determinada tecnologia a determinada actividade.

Neste contexto, e, infelizmente nas sociedades em que vivemos, a sinistralidade automóvel tem um peso demasiado elevado, tanto do ponto de vista humano como económico. É, pois, de importância vital que a assistência médica à sinistralidade tenha ao seu dispor ferramentas para prestar um melhor serviço às pessoas que dele necessitem.

Foi com este espírito de inovação e preocupação pelo bem-estar da sociedade que, depois de algumas reuniões de orientação, se tornou claro o grande benefício que poderia advir para a actividade dos serviços de emergência médica, a criação de uma nova ferramenta de auxílio. Efectivamente verificamos a existência de uma grave lacuna relacionada com o facto de que, quando um agente destes serviços chega a um local para prestar assistência não tem como obter dados biomédicos das pessoas a socorrer, especialmente nas situações em que estas se encontram inconscientes ou incapazes de comunicar. Foi então que, para resolver este problema, se idealizou o SEB.

2.1 Tecnologias

Existem vários tipos de tecnologias de comunicação sem fios que podem ser usadas para implementação da solução, mas entendemos que o Bluetooth é a mais eficaz, segura e robusta.

Apesar da tese ter como requisito a utilização do Bluetooth, achamos importante olhar para outras tecnologias, para que a sua escolha tenha bases de sustentação.

Assim desde logo tecnologias como o GSM (Global System for Mobile Communications) [11] e UMTS (Universal Mobile Telecommunications System) [7], são tecnologias que estão dependentes de infra-estruturas dedicadas e que, precisam de um operador de redes móveis para trabalhar. Estas limitações trazem problemas em situações de catástrofes naturais ou em cenários de bloqueio/congestão das redes infra-estruturadas em que as estruturas podem estar danificadas ou inoperacionais. Deste modo, é preferível a utilização das tecnologias que funcionam em modo Ad-Hoc (IEEE 802.11/Bluetooth), na medida em que não têm qualquer uma destas limitações, podendo funcionar de forma autónoma.

2.1.1 IEEE 802.11

Uma outra tecnologia possível de utilizar é a baseada no IEEE 802.11 (vulgarmente designada por WI-FI) que, no modo Ad-Hoc (ligação entre dois pontos), poderia ser uma solução viável para a implementação do sistema. Contudo, o Bluetooth (que opera na mesma banda de frequências do 802.11, os 2,4GHz) permite uma comunicação mais robusta, e disponibiliza 79 canais para comunicar [10], enquanto que o WI-FI, dispõe apenas de 14 [6]. Existe ainda o problema, dos dispositivos 802.11 ligados entre si, comunicarem sempre no mesmo canal, potenciando a possibilidade de interferências geradas por outras redes 802.11 que se encontrem no mesmo local.

2.1.2 Bluetooth

Na tecnologia Bluetooth, a comunicação é efectuada aplicando uma técnica de espalhamento espectral com saltos de frequência, denominada de FHSS (Frequency Hopping Spread Spectrum). Desta forma a frequência nunca é a mesma minimizando as interferências e possibilitando a existência de outras "redes" Bluetooth na mesma área geográfica.

Numa ligação Bluetooth, os dispositivos mudam de canal todos os 625μs, ou seja, 1600 vezes por segundo, sendo esta mudança feita de uma forma pseudo-aleatória [10].

2.2 Linux - Kurumin 7

O sistema operativo Linux foi eleito como uma das plataformas para a aplicação desenvolvida, pela sua filosofia *open-source*, o que permitiu uma maior flexibilidade e a possibilidade de alteração do seu código fonte. Na eventualidade de no futuro o SEB ser comercializado, os custos de produção serão inferiores uma vez que é um sistema operativo gratuito.

A escolha da distribuição recaiu sobre o Kurumin (distribuição de origem brasileira). A principal razão para a sua escolha foi a forma intuitiva com que esta possibilita a activação dos serviços relativos ao Bluetooth, o que permitiu fazer a aprendizagem de uma forma mais clara.

2.3 Cenários de Actuação

Este serviço foi idealizado para qualquer situação onde se encontre um ou mais veículos que estejam envolvidos num acidente de viação grave. Entenda-se por acidente grave, qualquer acidente que faça accionar os dispositivos de segurança, tais como os *airbags*. Caso o veículo não possua este tipo de dispositivos, o SEB poderá ser accionado por outro tipo de meio, como, por exemplo, um sensor piezoeléctrico.

Outros tipos de cenários possíveis são: o de um veículo que se precipita numa falésia; que fica oculto devido a algum tipo de deslocação de terras ou que depois de um despiste fica escondido por densa vegetação. Este tipo de acidentes podem provocar no veículo vários estragos que poderão activar o SEB, fazendo com que o veículo possa ser encontrado com mais facilidade pelos serviços de emergência, ou pelas autoridades que estejam na sua busca, principalmente em situações de emergência nocturnas.

2.3.1 Informação Necessária

O sistema tem de possuir a informação biomédica dos ocupantes dos veículos, para funcionar correctamente. Essa informação deverá ser introduzida pelos proprietários dos mesmos ou por pessoas que fiquem responsáveis por essa tarefa.

A informação deverá ser actualizada sempre que algum item seja alterado de forma significativa.

Os dados necessários a inserir no sistema são: **fotografia; nome; morada; estado civil; idade; peso; tipo de sangue; valor do colesterol; valor dos triglicerídeos; número de AVC (Acidente Vascular Cerebral); número de enfartes do miocárdio; valor da tensão arterial; valor das diabetes; se é fumador; alergias; medicação que está a tomar; outras doenças.** Cada ocupante do veículo terá o seu perfil com todos estes dados.

As fotografias dos passageiros devem ser inseridas no sistema através do interface USB (Universal Serial Bus) do mesmo, usando uma *Pen Drive* contendo na raiz uma pasta com o nome "fotos" contendo as fotografias. Estas deverão ser do tipo JPEG (Joint Photographic Experts Group) e com um nome padrão do tipo "pessoa1.jpg, pessoa2.jpg, pessoa3.jpg,….,pessoaN.jpg", conforme se trate do condutor habitual, não habitual, passageiro opcional 1, 2, 3,…N. A variável "N", neste e nos ficheiros do mesmo género, representa o número atribuído a cada passageiro. O tamanho de cada uma deverá respeitar a seguinte resolução: 200 pixeis de altura por 140 pixeis de largura.

2.3.2 Disponibilização da Informação

A informação do sistema está sempre disponível para ser consultada e actualizada pelas pessoas autorizadas.

Quando o sistema é activado, a informação é enviada em canal codificado, para as viaturas de emergência médica, que se aproximem do local e que a solicitem através do SEB.

2.3.3 Viaturas de Emergência Médica

Esta dissertação entende por viaturas de emergência médica todas aquelas que pertençam à entidade autorizada de cada pais a prestar este género de assistência. Em Portugal, essa entidade é o INEM (Instituto Nacional de Emergência Médica) que tem por missão garantir aos sinistrados ou vítimas de doença súbita a pronta e correcta prestação de cuidados de saúde [9].

2.4 Entidades que podem usufruir do SEB

O SEB encontra-se adaptado, de forma, a que as entidades de emergência médica e segurança (forças policiais, corporações de bombeiros, e protecção civil) possam usufruir dele, e eventualmente interagir com o SIRESP (Sistema Integrado das Redes de Emergência e Segurança de Portugal), actualmente regulado pela ANACOM (Autoridade Nacional de Comunicações) [2].

Muitas das vezes, senão todas, quem chega primeiro ao local de um sinistro automóvel são as forças de segurança que fiscalizam e controlam o tráfego. Assim sendo, se elas tiverem um conhecimento clínico prévio das pessoas que se encontram em dificuldades, poderão ir tomando algumas atitudes pró activas, enquanto não chegam os serviços médicos para efectuar a devida estabilização das vítimas.

Capítulo 3

Arquitectura do Sistema e Abordagem Técnica

3.1 Arquitectura do Sistema

3.2 Requisitos do Sistema

3.3 Pilares do Sistema

3 ARQUITECTURA DO SISTEMA E ABORDAGEM TÉCNICA

O SEB consiste na implementação de um sistema cliente-servidor que pode ser implementado em qualquer veículo automóvel.

O cliente é implementado nos veículos civis e faz o envio da informação biomédica relativa aos ocupantes dos mesmos. O servidor é implementado nos veículos de assistência e emergência médica, ou autoridades, e recebe do cliente a informação que lhe é enviada, de modo a que os profissionais de saúde possam consultar a informação referida nas suas viaturas.

Em termos práticos, o sistema funciona da seguinte maneira: quando um veículo sofre um acidente grave, o SEB, em modo cliente, é activado automaticamente, e o cliente entra de imediato no modo de procura até encontrar um veículo dotado do SEB, em modo servidor. No momento em que o encontra, envia, através da comunicação rádio Bluetooth, a informação biomédica de cada ocupante. Deste modo, as equipas de emergência têm acesso a informação importante, para o desempenho das suas funções, de forma a, prestar uma assistência às vítimas, mais rápida, segura e eficaz.

3.1 Arquitectura do Sistema

A Figura 3.1 mostra as camadas protocolares onde o sistema SEB assenta.

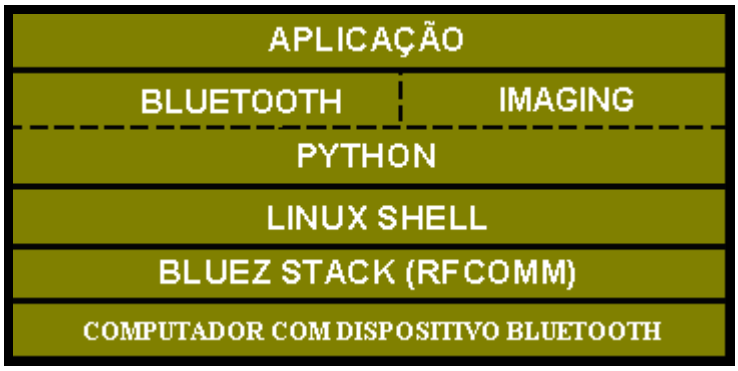

Figura 3.1: Camadas Protocolares do sistema SEB

Do nível físico para o nível da aplicação temos:

- **"COMPUTADOR COM DISPOSITIVO BLUETOOTH",** é a base, o *hardware*, onde todos os elementos se apoiam e que possuiu um elemento fundamental, que permite a comunicação ao nível físico, o dispositivo Bluetooth.

- "**BLUEZ STACK (RFCOMM)**", é a pilha de protocolos integrados nas versões do Linux Kernel v2.4 e v2.6 que permitem ao sistema operativo Linux fazer a gestão dos dispositivos Bluetooth [4]. A BlueZ Stack integra o protocolo RFCOMM (Radio Frequency Communication) que, por sua vez, permite realizar a comunicação, via a emulação de portas série. É a partir da emulação destas portas que são criados os *sockets*, onde se realiza a comunicação entre o cliente e o servidor.

- Nos dois patamares seguintes, estão as linguagens de programação usadas no desenvolvimento da aplicação de gestão do sistema e no envio de dados por Bluetooth.

  Assim, para criar o *front-end* do sistema que serve de interface com o utilizador optou-se pela linha de comandos do sistema operativo, a "**LINUX SHELL**" [5], por esta ser a sua linguagem nativa. Quanto ao "**PYTHON"** [8], foi escolhido depois de uma criteriosa pesquisa que revelou ser esta uma das melhores ferramentas para desenvolver aplicações que têm por base o Bluetooth. Esta linguagem de programação possui dezenas de módulos, dos quais são usados dois específicos, o *Bluetooth (PyBlueZ)* [1] e o *Imaging*. O primeiro, permite que o Python fique dotado com um conjunto de classes e funções úteis para as mais comuns tarefas de programação relacionadas com o Bluetooth, tais como a criação de *sockets*. O último, é usado para dotar o Phyton da capacidade de processamento de imagens, necessária para fazer o processamento das fotografias dos ocupantes das viaturas.

- "**APLICAÇÃO",** é a última camada e refere-se ao *software* que gere todo o sistema, incluindo a interface com os utilizadores, permitindo que tudo que esteja relacionado com a introdução, alteração, eliminação, envio e recepção de dados, entre outras funções, seja devidamente efectuado.

3.2 Requisitos do Sistema

3.2.1 Sistema Informático/*Hardware*

Cada veículo deve possuir um módulo informático, que deverá estar integrado com a viatura, dotado do sistema operativo referido na Secção 2.2 e com os módulos da linguagem de programação Python*, Bluetooth* e *Imaging* instalados. Devem, ainda, estar equipados com um dispositivo Bluetooth*,* cuja antena deve ser colocada no exterior do habitáculo do veículo para que o sinal Bluetooth seja enviado e recebido com o mínimo de interferências e o máximo de potência. Estes requisitos devem-se observar nas partes cliente e servidor.

3.2.2 Ligação do Sistema aos Veículos

É necessário que a ligação entre o SEB, e os veículos com ele equipados, seja feita de forma, a que o sistema seja activado apenas nos momentos certos. Nesta fase não, existe o objectivo de aprofundar a forma como essa ligação será feita, por isso, caberá, no futuro, aos construtores dos veículos automóveis e à entidade que eventualmente implemente o serviço, procurar uma resposta para essa questão. Contudo, parece lógico que tornar este sistema solidário com o sistema de segurança activa do veículo poderá ser uma solução para o problema. Um exemplo disso poderá ser uma ligação *ON-OFF* com o *airbag* do condutor que, em caso de abertura, será o suficiente para accionar também o sistema SEB.

3.2.3 Ocupantes por Veículo

O número de ocupantes por veículo é ajustável, ou seja, é possível definir na aplicação de gestão do sistema, no cliente, a lotação do veículo.

O sistema prevê a criação de um perfil para o condutor habitual, para um condutor não habitual e para cada um dos restantes passageiros (passageiros adicionais).

3.3 Pilares do Sistema

O sistema está dividido em duas partes distintas, o cliente e o servidor, que partilham o mesmo *hardware*. Mas nas camadas "**APLICAÇÃO", "PYTHON (*BLUETOOTH, IMAGING*)", "LINUX SHELL"** e **"BLUEZ STACK (RFCOMM)"** existem diferenças, que serão explicadas em pormenor no quarto capítulo, pois uma parte envia e a outra recebe a informação.

Para que o cliente tenha informação para enviar ao servidor, é necessário que esta seja introduzida na base de dados do sistema através da aplicação de gestão, que passa a ser descrita na Subsecção 3.3.1.

3.3.1 Aplicação de Gestão, Parte Cliente

Esta aplicação foi desenvolvida para que, através dela, se possa fazer a gestão de todo o sistema no que se refere à introdução, eliminação, verificação e alteração da informação de cada ocupante, de cada veículo, e também a activação manual do SEB.

Para impedir que o acesso à aplicação seja feito por pessoas não autorizadas, incluindo as crianças que com a sua curiosidade natural poderiam alterar os dados existentes na base de dados do sistema, foi criado um mecanismo de segurança que, sempre que é tentada a abertura da aplicação, esta pede a inserção de um código de segurança para continuar, tal como se pode ver na Figura 3.2.

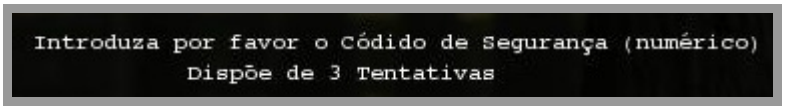

Figura 3.2: Inserção do Código de Segurança

A aplicação de gestão do sistema pode ser vista na Figura 3.3. A ordem das opções corresponde à ordem pela qual a informação deverá ser inserida.

Figura 3.3: Aplicação do Sistema Parte Cliente, embutida na Consola dos Veículos Civis

3.3.1.1 Opção 1 - Inserir Matrícula e Número Máximo de Passageiros

Em primeiro lugar, é necessário definir a matrícula e a lotação do veículo. A matrícula é associada ao perfil de cada passageiro para que, em situações em que vários veículos são envolvidos num acidente, se possa diferenciar quem pertence a cada um.

É também necessário definir o número máximo de passageiros, para se estabelecer o limite máximo de perfis a inserir na base de dados, tal como se vê na Figura 3.4.

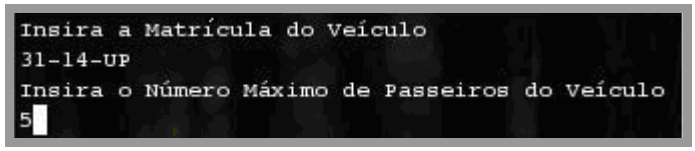

Figura 3.4: Inserção da Matrícula e Lotação

3.3.1.2 Opção 2 - Inserir Fotos dos Passageiros

O segundo passo na inserção de dados deverá ser a cópia das fotografias dos passageiros que vão ser introduzidos no sistema, cujo procedimento de inserção já foi explicado na Subsecção 2.3.1.

3.3.1.3 Opção 3 - Inserir Dados do Condutor Habitual

Esta opção permite criar o perfil do condutor habitual do veículo com todos os dados já referidos na Subsecção 2.3.1.

3.3.1.4 Opção 4 - Inserir Dados de um Condutor Não Habitual

Da mesma forma da opção anterior, esta permite inserir os dados de outra pessoa que esteja autorizada a conduzir o veículo, por exemplo um parente próximo.

3.3.1.5 Opção 5 - Inserir Dados dos Passageiros

Esta opção está dividida em sub opções, uma para cada um dos passageiros que não estão incluídos na opção 3 ou 4 do "Menu Principal", tal como se pode ver na Figura 3.5, que apresenta o "Menu Auxiliar - Inserção" para o caso de uma viatura com lotação de cinco passageiros.

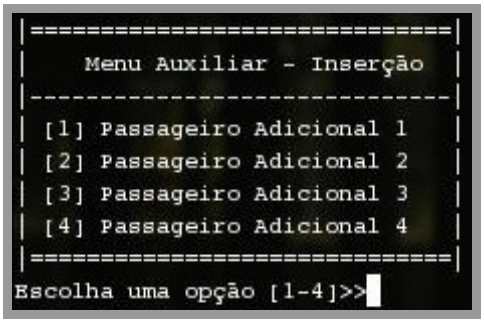

Figura 3.5: Menu Auxiliar - Inserção

3.3.1.6 Opção 6 - Consultar Informação da Base de Dados

Esta opção pode ser utilizada para consultar a informação existente na base de dados do sistema, relativa a cada passageiro do veículo, individualmente ou na totalidade, tal como se pode ver na Figura 3.6. A informação relativa à matrícula e à lotação é disponibilizada no cabeçalho do "Menu Auxiliar - Visualização".

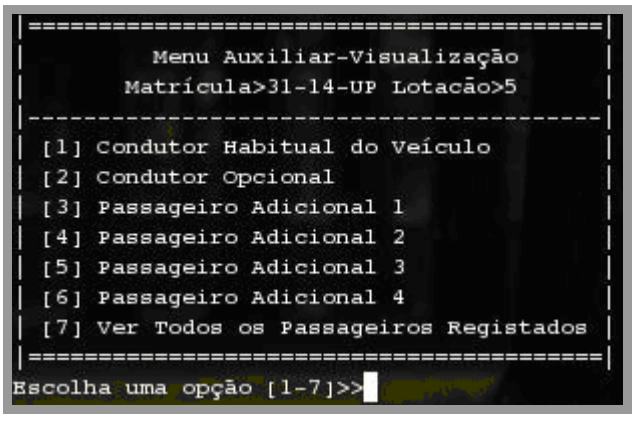

Figura 3.6: Menu Auxiliar - Visualização

3.3.1.7 Opção 7 - Apagar Perfis Existentes

A opção número 7 serve para eliminar os perfis dos passageiros existentes na base de dados. As opções são disponibilizadas pelo "Menu Auxiliar - Eliminação" (Figura 3.7).

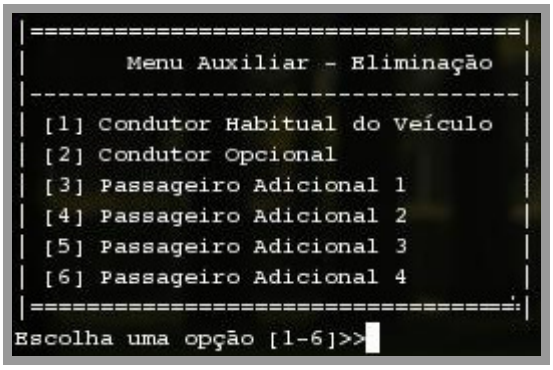

Figura 3.7: Menu Auxiliar - Eliminação

3.3.1.8 Opção 8 - Activar Serviço de Emergência

Esta opção foi criada para efeito de testes e para situações de emergência em que os passageiros podem activar o sistema manualmente. Ao escolher esta opção, o sistema é activado e entra em modo de procura por meios de socorro com o SEB activo, tal como se pode verificar na Figura 3.8.

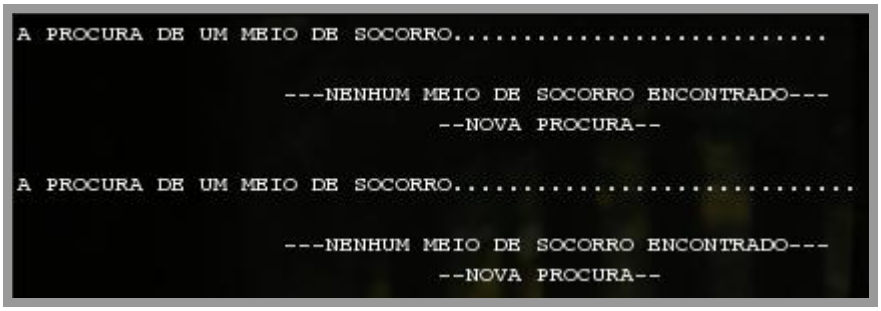

Figura 3.8: Procura por Meios de Socorro

Quando é encontrado um meio de socorro, a informação biomédica é enviada para ele e exibida a mensagem da Figura 3.9 no veículo sinistrado.

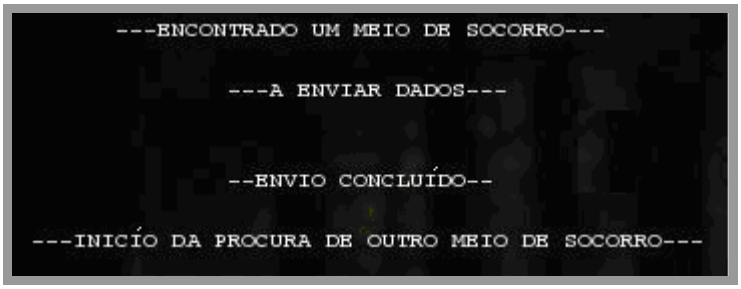

Figura 3.9: Encontro do Meio de Socorro e Envio de Dados

No fim de cada transmissão, o sistema continua em modo de procura por mais meios de socorro, até ser desligado manualmente. O objectivo de ter sempre o sistema em modo de procura passa por possibilitar o envio da informação biomédica para mais do que um tipo de meio de socorro que chegue ao local do acidente.

3.3.1.9 Opção 9 - Alterar Código de Segurança

Esta opção destina-se à alteração do código que permite o acesso à aplicação do sistema.

3.3.1.10 Opção 10 - Sair

A última opção deverá ser utilizada para sair da aplicação.

3.3.2 Aplicação de Gestão, Parte Servidor

A aplicação de gestão, na parte servidor, pode ser vista na Figura 3.10 e dispõe de quatro opções.

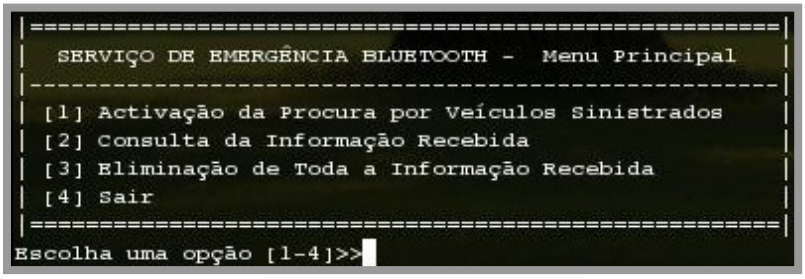

Figura 3.10: Aplicação do Sistema, Parte Servidor

3.3.2.1 Opção 1 - Activação da Procura por Veículos Sinistrados

A primeira opção tem como função colocar o servidor em modo de procura, por um ou mais clientes, ou seja, esta opção deverá ser seleccionada quando um meio de socorro parte ao encontro dos veículos sinistrados.

Na Figura 3.11, podemos ver o SEB do meio de socorro em modo de procura e, no momento em que encontra o veículo sinistrado, a receber a informação biomédica dos seus passageiros.

Figura 3.11: Procura por Veículos Sinistrados e Recepção da Informação

A aplicação no servidor, quando recebe a informação biomédica dos passageiros dos veículos sinistrados, apresenta-a e, ao mesmo tempo guarda-a para análise posterior.

Sempre que a informação biomédica, relativa aos passageiros de um veículo é recebida, é verificado se ela já existe no sistema. Caso isso se confirme, ela não volta a ser apresentada nem guardada. Nesta situação, a aplicação do sistema apresenta a mensagem que se pode ver na Figura 3.12.

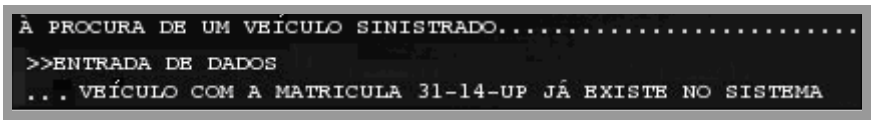

Figura 3.12: Mensagem apresentada quando a Informação Biomédica já existe no Sistema

3.3.2.2 Opção 2 - Consulta da Informação Recebida

A informação guardada pode ser consultada a partir da opção número dois da aplicação.

Na Figura 3.13 pode ver-se o menu apresentado, aquando da escolha dessa opção.

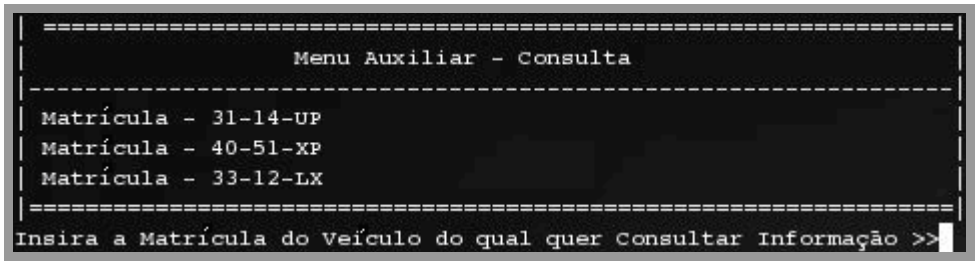

Figura 3.13: Menu Auxiliar - Consulta

Tal como se pode verificar, o menu auxiliar apresenta como escolha três matrículas diferentes, o que significa que o SEB do veículo de emergência já recebeu a informação biomédica dos ocupantes destas três viaturas.

3.3.2.3 Opção 3 - Eliminação de Toda a Informação Recebida

Esta opção possibilita a eliminação de toda a informação presente no sistema da parte servidor, isto porque, depois de processada e analisada, não faz sentido que ela continue no sistema do veículo de emergência.

3.3.2.4 Opção 4 - Sair

A última opção deverá ser usada para sair da aplicação do sistema.

Capítulo 4

Desenvolvimento e Implementação do Sistema

4.1 Armazenamento da Informação

4.2 Tratamento da Informação

4.3 Comunicação entre Entidades

4.4 Envio e Recepção de Dados

4.5 Aplicação do Sistema, Entidade Cliente

4.6 Aplicação do Sistema, Entidade Servidor

4 DESENVOLVIMENTO E IMPLEMENTAÇÃO DO SISTEMA

Neste capítulo é descrito o método de desenvolvimento do *software* da aplicação do sistema.

Nas Secções 4.5 e 4.6, a aplicação do sistema é descrita na sua totalidade, sendo referido, em alguns casos, algumas linhas de código, ou noutros mais relevantes, a sua totalidade, havendo lugar à sua explicação.

4.1 Armazenamento da Informação

Quando o sistema foi idealizado, sabia-se, à partida, que teria de haver uma forma de guardar a informação biomédica. Chegou-se à conclusão que não havia necessidade de se recorrer a um modelo de base de dados tradicional. Verificou-se ser mais eficiente guardar a informação biomédica de cada passageiro em ficheiros de texto (com a excepção das fotos) com o nome "perfilN.txt". Cada linha destes ficheiros contém os dados referidos na Subsecção 2.3.1 e pode-se verificar um exemplo na Figura 4.1.

Todos os dados biomédicos presentes na dissertação, com a excepção dos nomes e das fotografias apresentadas, são fictícios.

```
«««««« PASSAGEIRO ADICIONAL 2 »»»»»» 31-14-UP ««««»»»»
Ricardo Cruz
Pedrouços, Maia
Estado Civil-Solteiro
Idade-23
Peso-80
Tipo de Sangue-B+
Colesterol-190
Triglicerídeos-120
AVC-0
Enfartes-0
Tensão Arterial-150/70
Diabetes-400
Fumador-N
Alérgico A-Nada
Medicação-Nada
Outras Doenças-Nada
```

Figura 4.1: Exemplo de Conteúdo dos Ficheiros PerfilN.txt

As fotos de cada passageiro, encontram-se noutro tipo de ficheiros, com o nome "pessoaN.jpg".

A matrícula e a lotação de cada veículo são também armazenadas em ficheiros de texto, "pamatricula.txt" e "maxocup.txt", respectivamente.

O código de entrada na aplicação é por defeito "1111" e fica armazenado no ficheiro "pin.txt". Este código pode ser alterado a qualquer momento.

Todos os ficheiros, incluindo os da aplicação do sistema na entidade cliente e servidor, são guardados no sistema operativo, no directório "/seb". As fotografias dos passageiros na entidade cliente são guardadas no directório "/seb/fotos". Quanto à informação recebida pela entidade servidor, ela é guardada no directório "/seb/*matricula*", onde "matricula" representa as matrículas das viaturas, das quais a entidade servidor recebe a informação.

Este tipo de armazenamento provou ser uma forma segura de organizar e guardar a informação, o que tornou a sua gestão simples e eficaz.

4.2 Tratamento da Informação

Para que se obtenha o resultado que se vê na Figura 4.2, os dados, depois de inseridos, precisam de ser "trabalhados".

Figura 4.2: Imagem Final, pronta para envio, relativa ao Passageiro Opcional 2

Este resultado é conseguido através de um módulo de *software* do sistema, desenvolvido em Python, chamado "IMAGE.PY" (explicado na Subsecção 4.5.2.4), e tem como objectivo processar as imagens dos passageiros. Em primeiro lugar, é criada uma imagem com fundo cinzento, com 380x260 pixeis de tamanho; de seguida, a fotografia do passageiro é sobreposta sobre essa imagem e, finalmente, a informação biomédica de cada passageiro, contida no ficheiro de texto associado ao seu perfil, é colocada sobre essas duas imagens em locais predefinidos, de forma a criar a imagem final, que se vê na Figura 4.2, e que é exibida na aplicação das viaturas de emergência médica ou autoridades.

As imagens finais correspondentes a cada passageiro são criadas com o nome "passageiroN.jpg" e são enviadas para um ficheiro compactado chamado "media.tgz", no momento anterior ao envio do mesmo, de modo a que a informação enviada seja sempre a mais actual.

A entidade servidor, assim que recebe o ficheiro "media.tgz", descompacta os ficheiros que este contém e apresenta as imagens finais de cada passageiro.

4.3 Comunicação entre Entidades

Um dos primeiros objectivos a atingir neste trabalho foi o de tornar possível a troca de informação entre duas entidades Bluetooth desconhecidas entre si, sendo essa informação texto ou imagem e não havendo limite quanto à quantidade. A solução encontrada foi fazer o envio da informação através do protocolo Bluetooth RFCOMM que permite a comunicação através da criação de canais (até 60 em simultâneo) fazendo a emulação de portas série.

Desenvolver aplicações para Bluetooth em Python significa ter que programar utilizando o modelo dos *sockets*. O *socket* representa uma extremidade do canal de comunicação e, neste modelo, é criado utilizando o protocolo RFCOMM, estabelecendo uma linha de comunicação série emulada que permite a troca de informação.

Como existem vários cenários possíveis de comunicação, as entidades que vão comunicar, cliente e servidor, são sempre diferentes. Assim, os endereços Bluetooth das duas partes não são conhecidos, por isso é necessário utilizar o protocolo Bluetooth SDP (Service Discovery Protocol), na entidade servidor, para anunciar o serviço e, na entidade cliente, para o procurar. Esse anúncio fornece o canal e endereço Bluetooth para o *socket* RFCOMM do cliente, de modo a estabelecer-se a comunicação entre as duas partes.

4.4 Envio e Recepção de Dados

Para fazer o envio e recepção de dados foram desenvolvidos dois módulos de *software*: o "RFCOMM-CLIENT.PY" (explicado na Subsecção 4.5.3) implementado na entidade cliente, para a enviar a informação biomédica, e o "RFCOMM-SERVER.PY" (explicado na Subsecção 4.6.2) implementado na entidade servidor, para criar o serviço, que vai permitir receber essa informação.

Estando a ligação estabelecida é possível fazer o envio de dados. Como já escrito na Secção 4.2, o ficheiro a ser enviado do cliente para o servidor tem o nome "media.tgz".

Efectivamente as imagens finais dos passageiros têm que ser enviadas num só ficheiro, caso contrário seria necessário estabelecer uma conexão por cada ficheiro enviado, tornando o tempo de envio da informação mais longo e proporcional ao número de passageiros de cada veículo.

4.5 Aplicação do Sistema, Entidade Cliente

Esta secção pretende dar a perceber ao leitor como é que a aplicação do sistema foi desenvolvida ao nível da programação, que foi baseada na Linux Shell e na linguagem Python.

4.5.1 Validação do Código de Entrada

A validação da entrada na aplicação é logo executada no início e o utilizador dispõe de três tentativas.

O código de entrada por defeito é estabelecido na altura da implementação. Deverá ser numérico e o número de dígitos não está limitado.

Segue-se um resumo na Figura 4.3 da função que implementa a validação.

```
valida ()
{
 i=1
 j=3
 clear
 if [ $i -le 3 ] ; then
  .
  .
  .
  echo "   Introduza por favor o código de validação (numérico)"
  echo "                Dispõe de $j tentativas"
  let j=$j-1
❶ read codigo
❷ c=$(cat pin.txt)
❸ if [ $codigo -eq $c ] ; then menu
   else
        let i=$i+1
        echo ERRADO,TENTE DE NOVO
        valida
  fi
 else exit 0
 fi
}
```

Figura 4.3: Função de Validação do Código de Segurança

Na figura acima, e nas seguintes, as bolas azuis representam as linhas mais relevantes das funções.

Passo a descrevê-las: (linha ❶) é pedido ao utilizador para inserir o código de entrada e este é lido para a variável "codigo"; (linha ❷) é passado para a variável "c" o valor do código da aplicação; na linha ❸ é comparado o código introduzido com o código da aplicação. Se os códigos forem iguais é chamada a função "menu()" que dá acesso ao "Menu Principal", senão é incrementada a variável "i" e chamada novamente a função "valida()" até que se esgote o número máximo de tentativas estabelecido.

4.5.2 Menu Principal e Introdução de Dados

Tendo sucesso a validação do código de entrada, aparece o "Menu Principal" que faz a gestão da aplicação do sistema, implementado pela função "menu()". Na Figura 4.4 pode-se ver uma parte da mesma.

```
menu ()
{
while :
 do
     clear
     echo "|=========================================================|"
     echo "|  SERVIÇO DE EMERGÊNCIA BLUETOOTH -  Menu Principal      |"
     echo "|---------------------------------------------------------|"
     echo "| [1] Inserir Matrícula e Número Máximo de Passageiros    |"
     echo "| [2] Inserir Fotos dos Passageiros                       |"
     echo "| [3] Inserir Dados do Condutor Habitual                  |"
     echo "| [4] Inserir Dados de um Condutor Não Habitual           |"
     echo "| [5] Inserir Dados dos Passageiros                       |"
     echo "| [6] Consultar Informação da Base de Dados               |"
     echo "| [7] Apagar Perfis Existentes                            |"
     echo "| [8] Activar o Serviço de Emergência                     |"
     echo "| [9] Alterar Código de Segurança                         |"
     echo "| [10] Sair                                               |"
     echo "|=========================================================|"
     echo -n "Escolha uma opção [1-10]>>"
  ❹  read yourch
  ❺  case $yourch in
       1) clear
  ❻       matricula
          ;;
       2) clear
          echo "COLOQUE A PEN DRIVE COM AS FOTOS A INSERIR, NO DIRECTÓRIO /fotos"
          echo "CARREGUE EM ENTER QUANDO TIVER CONCLUÍDO ESTA OPERAÇÃO"
          read
  ❼       sudo mount /dev/sda1 /seb/pendrive
  ❽       cp /seb/pendrive/fotos/*.jpg /seb/fotos
          echo "FOTOS INSERIDAS, PODE RETIRAR A PEN DRIVE"
  ❾       sudo umount /dev/sda1
          sleep 2
          ;;
       3) clear
  ❿       existe 1
  ❶❶     mat=$(cat pamatricula.txt)
  ❶❷     echo  ««  CONDUTOR HABITUAL  »»   $mat   «»» > perfil1.txt
  ❶❸     insere 1
      ;;
...
}
```

Figura 4.4: Função que implementa o Menu Principal da Aplicação do Sistema, Entidade Cliente

São apresentadas as opções disponíveis e é solicitado ao utilizador que escolha uma delas. A leitura da escolha é feita na linha ❹ para a variável "yourch" e, logo a seguir na linha ❺, é verificada qual a opção escolhida.

4.5.2.1 Matrícula e Lotação dos Veículos

As opções de inserção de dados no sistema são cinco, sendo conveniente introduzir a matrícula e lotação do veículo antes de qualquer outra informação, para que a matrícula seja associada ao perfil dos passageiros e para se definir o número opções nos menus auxiliares.

Se a opção escolhida for a número 1 é de imediato chamada na linha ❻ a função "matricula()", que se vê de seguida na Figura 4.5.

```
matricula()
{
    echo "Insira a Matrícula do Veículo"
    read matricula
❶❹ echo $matricula > pamatricula.txt
    echo "Insira o Número Máximo de Passeiros do Veículo"
    read maxocup
❶❺ echo $maxocup > maxocup.txt
    echo $maxocup > pamaxocup.txt
    a=1
    let maxocup=$maxocup+1
    rm -fr fotos/pess*.jpg
    rm -fr passagei*.jpg
    rm -fr perfil*.txt
    while [ $a -le $maxocup ]
    do
❶❻  echo  ESTE PASSAGEIRO NÃO ESTÁ REGISTADO > perfil$a.txt
     let a=$a+1
    done
   return
 }
```

Figura 4.5: Função de Inserção de Matrícula e Lotação dos Veículos

Esta função recebe do utilizador a informação referida e coloca-a nos ficheiros de texto respectivos, como se vê nas linhas ❶❹ e ❶❺. Sabendo-se o número máximo de passageiros, o ficheiro de perfil de cada um é de imediato criado, contendo na primeira linha o texto "ESTE PASSAGEIRO NÃO ESTÁ REGISTADO", tal como se vê na linha ❶❻. Este texto é

útil para se poder aferir se determinado passageiro já se encontra no sistema, como se pode verificar na função "existe()", descrita na Subsecção 4.5.2.3.

4.5.2.2 Fotos dos Passageiros

Deve-se iniciar a introdução dos dados biomédicos dos passageiros, logo após a definição da matrícula e do número máximo de passageiros de um veículo. Essa inserção de dados deve começar pelas fotografias.

Esta tarefa é implementada pela aplicação usando alguns dos comandos que se podem ver na Figura 4.4. O primeiro passo consiste em "montar" a *Pen Drive* com as fotos dos passageiros (linha ❼), seguindo-se a copia dos ficheiros no formato JPEG que estão no directório "/fotos" da *Pen Drive*, para o directório "/seb/fotos" da aplicação do sistema (linha ❽) e termina com o "desmontar" da *Pen Drive* (linha ❾). Desta forma, a aplicação do sistema está pronta para continuar a inserção dos dados biomédicos.

4.5.2.3 Condutor Habitual

Sempre que são introduzidos dados de um passageiro, é verificado se o mesmo existe no sistema, de forma a prevenir uma dupla inserção ou a alteração dos dados já existentes. Para que tal aconteça, foi criada a função "existe()", que é chamada na linha ⑩ (Figura 4.4) com o valor "1" como argumento, para que, neste caso em concreto, verifique se o perfil do condutor habitual do veículo já foi criado. Pode-se ver esta função na Figura 4.6.

```
existe()
{
⑰ k=$(cat perfil$1.txt)
⑱ if [ "$k" == "ESTE PASSAGEIRO NÃO ESTÁ REGISTADO" ]; then echo
     else
⑲    echo "ESTE PERFIL JÁ ESTÁ REGISTADO, DESEJA ALTERÁ-LO (S/N)"
     read escolha
⑳    if [ $escolha == S ]; then echo
       else
        menu
     fi
   fi
}
```

Figura 4.6: Função que verifica a existência ou não de Passageiros Registados

A função trabalha da seguinte forma: na linha ⑰ a variável "k" fica com o valor do conteúdo do ficheiro que guarda o perfil do passageiro. Se o ficheiro contiver o texto "ESTE PASSAGEIRO NÃO ESTÁ REGISTADO" (linha ⑱), a aplicação dá início à inserção de dados. Caso o texto presente no ficheiro não seja o referido, a aplicação alerta o utilizador de que o perfil já se encontra preenchido e "pergunta-lhe" se deseja prosseguir (linha ⑲). Se a resposta for afirmativa (linha ⑳), a aplicação prossegue para as linhas de código seguintes que levam à inserção de dados, senão, volta ao "Menu Principal".

Do início do processo de inserção de dados faz parte a linha ⑪ (Figura 4.4) que associa a variável "mat" à matrícula do veículo, inserida no sistema pela função "matricula()". A linha ⑫, da mesma figura, faz com que a primeira linha do ficheiro de perfil de passageiro contenha a informação relativa ao papel do passageiro na viatura e, a matrícula da mesma, à qual ele pertence, para que, em casos de acidentes que envolvam vários veículos, se possa fazer a associação entre passageiros e viaturas. Na linha imediatamente a seguir (linha ⑬)

também da mesma figura, é chamada a função "insere()" responsável pela inserção dos dados biomédicos dos passageiros, e que se pode ver um pequeno excerto na Figura 4.7. A esta função é-lhe colocada como parâmetro o número 1, pois a inserção de dados vai ser relativa ao condutor habitual do veículo.

```
insere ()
{
.
.
.
      ❷❶    echo "Diabetes (Valor/N):"
        ""    read diabetes
        ""    echo Diabetes-$diabetes >> perfil$1.txt
        ""    echo "Fuma (S/N):"
        ""    read fuma
        ""    echo Fumador-$fuma >> perfil$1.txt
        ""    echo "Alergias:"
        ""    read alergias
        ""    echo Alergico A-$alergias >> perfil$1.txt
.
.
.
     ❷❷    python image.py fotos/pessoa$1.jpg perfil$1.txt passageiro$1.jpg &
}
```

Figura 4.7: Imagem parcial da Função de Inserção de Dados Biomédicos

A linha ❷❶ representa uma parte da função. Nela se verifica que é pedido ao utilizador a inserção de três dados biomédicos: diabetes, se é fumador e alergias, e, logo de seguida, esses dados são guardados no seu ficheiro de perfil.

Já com todos os dados relativos a determinado passageiro inseridos, a função termina com a criação da imagem final "passageiroN.jpg" (linha ❷❷), recorrendo, para tal, ao módulo "IMAGE.PY" desenvolvido em Python, e que se pode ver na Figura 4.8.

4.5.2.4 Módulo "IMAGE.PY"

```
❷❸ import Image
"" import ImageDraw
   import sys

❷❹ im = Image.new("RGB", (380,260), "black")
   draw = ImageDraw.Draw(im)
❷❺ im.save("fotos/fundocinza.jpg", "JPEG")

❷❻ icon = Image.open(sys.argv[1])
❷❼ x, y = icon.size
❷❽ im.paste(icon, (0,0,x,y))
❷❾ im.save ("base.jpg", "JPEG")

❸❿ img = Image.open("base.jpg")
   draw = ImageDraw.Draw(img)

   p=0
   i=1
❸❶ f = open(sys.argv[2],"r")

❸❷ while i<4:
❸❸  texto=f.readline()
❸❹  pos = 4,200+p
❸❺  draw.text(pos, texto)
❸❻  img.save(sys.argv[3])
    i=i+1
❸❼  p=p+11

   a=0
❸❽ while i<18:
    texto=f.readline()
    pos = 145,0+a
    draw.text(pos, texto)
    img.save(sys.argv[3])
    i=i+1
    a=a+11
```

Figura 4.8: Módulo que faz o tratamento das Imagens dos Passageiros

Este módulo recorre à PIL (Phyton Imaging Library) para poder ter capacidades de processamento de imagens. A PIL possui vários módulos, mas para o "IMAGE.PY" apenas dois deles, o "*Image*" e o "*ImageDraw*", foram utilizados. Eles têm, respectivamente, a capacidade de criar imagens e de acrescentar algo a uma imagem.

Olhando para a programação verifica-se que se começa por importar os módulos já referidos (linha ❷❸).

O próximo passo é a criação de uma imagem com fundo cinzento, que será a base da imagem final, com 380 pixeis de largura por 260 de altura (linha ❷❹). Essa imagem é guardada para o ficheiro "fundocinza.jpg" (linha ❷❺).

De seguida, é aberta a imagem do ficheiro "pessoaN.jpg" que contém a foto de cada passageiro (linha ❷❻) e verificado o seu tamanho (linha ❷❼).

Na linha ❷❽, a foto do passageiro é colada por cima da imagem "fundocinza.jpg" e gravada sob o nome "base.jpg" (linha ❷❾); o resultado pode ser visto na Figura 4.9.

Figura 4.9: Exemplo da Imagem Base com a Fotografia de um Passageiro

Depois a imagem "base.jpg" é aberta e atribuída à variável "img" (linha ❸❿) para, de seguida, se iniciar o processo de sobreposição do conteúdo do ficheiro "perfilN.txt", que começa na linha ❸❶ com a atribuição dessa informação à variável "f". E continua com a implementação de dois ciclos "*while*". O primeiro, inicia-se na linha ❸❷ e executa o seguinte:

❸❸: na primeira iteração, a primeira linha do ficheiro "perfilN.txt" é atribuída à variável "texto", e assim sucessivamente até acabar o ciclo.

❸❹: a variável "pos" representa a posição em pixeis na imagem "base.jpg", onde deve ser colocado o conteúdo da variável "texto".

❸❺: o código presente nesta linha é responsável por "escrever", na imagem "base.jpg", e na posição "pos", o valor da variável "texto".

❸❻: a alteração à imagem inicial é gravada para o ficheiro que vai ser o *output* de todo o processo, o ficheiro "passageiroN.jpg".

❸❼: a incrementação da variável "p" é necessária para que o texto, que vai surgir na iteração seguinte, seja colocado na imagem, na linha seguinte.

O ciclo é executado durante três iterações, o que produz a imagem da Figura 4.10.

Figura 4.10: Continuação da construção da Imagem Final

O ciclo "*while*" seguinte (linha ❸❽) é idêntico ao primeiro, as diferenças essências são o número de iterações, que são dezoito, e as posições onde é escrito o texto restante do ficheiro "perfilN.txt". O resultado final é o presente na Figura 4.11.

Figura 4.11: Imagem final correspondente ao Condutor Habitual do Veículo

4.5.2.5 Condutor Não Habitual

Sendo a quarta opção do menu da aplicação do sistema, o processo de introdução de dados do condutor não habitual é em tudo idêntico ao do condutor habitual.

4.5.2.6 Passageiros

O número de passageiros de cada veículo depende da sua lotação. Para tornar o sistema flexível, a aplicação foi desenvolvida de modo a que as opções disponíveis no "Menu Auxiliar - Inserção", estejam directamente relacionadas com a informação inserida na opção 1 do "Menu Principal", ou seja, o número de opções varia conforme a lotação do veículo. Se, por exemplo, a lotação for de 9 passageiros, estarão disponíveis 8 opções, pois os dados do condutor habitual e não habitual são introduzidos à parte. Esta situação pode ver-se na Figura 4.12.

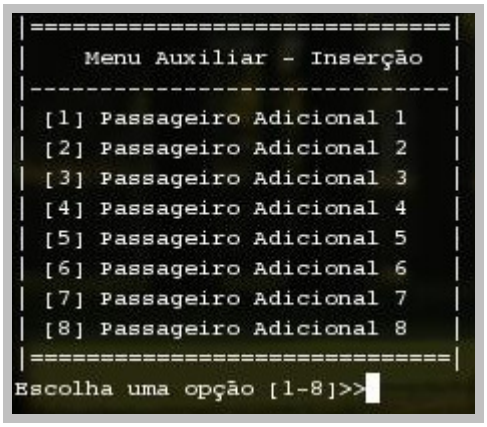

Figura 4.12: Menu Auxiliar - Inserção, para Lotação de Nove Passageiros

De seguida, na Figura 4.13, vê-se um excerto da implementação do "Menu Auxiliar - Inserção".

```
menu()
{
.
.
.
4)    clear
      f=1
      max=$(cat maxocup.txt)
      let max=$max-1
      echo "|============================|"
      echo "|    Menu Auxiliar - Inserção  |"
      echo "|----------------------------|"
 ❸❾   while [ $f -le $max ]
      do
 ❹❿     echo "| [$f] Passageiro Adicional $f   |"
        let f=$f+1
        done
        echo "|============================|"
        .
        .
        .
 ;;
.
.
.
}
```

Figura 4.13: Construção do Menu Auxiliar - Inserção

A construção do menu auxiliar tem de ser dinâmica, o que implica a implementação de um ciclo "*while*" (linha ❸❾) que crie nesse menu uma entrada/opção (linha ❹❵) para cada passageiro adicional.

4.5.2.7 Consulta da Informação

A sexta opção do "Menu Principal" recorre a um sub menu dinâmico com características idênticas ao sub menu "Menu Auxiliar - Inserção".

Esta opção serve exclusivamente para consultar a informação relativa aos passageiros e apresenta, também, a matrícula e lotação da viatura. A Figura 3.6 ilustra o "Menu Auxiliar - Visualização" para um veículo com lotação de 5 passageiros.

Quando é escolhida uma opção, ela é colocada como parâmetro à função "ver()" que de imediato é chamada (Figura 4.14).

```
ver ()
{
    a=1
    clear
    max=$(cat maxocup.txt)
   let s=$max+2
   let j=$max+1
    echo -n "/usr/bin/showfoto" > imagens.sh
❹❶ if [ $1 -eq $s ]; then
❹❷  while [ $a -le $j ]
     do
❹❸    echo -n " passageiro$a.jpg" >> imagens.sh
      echo -n "."
      let a=$a+1
     done
    chmod 777 imagens.sh
❹❹ /seb/imagens.sh &
    menu
   fi

❹❺ showfoto passageiro$1.jpg &
    return
}
```

Figura 4.14: Função que permite a consulta de dados da Base de Dados

A função verifica (linha ❹❶) se a opção escolhida foi a última do "Menu Auxiliar - Visualização", ou seja, a que permite consultar os dados de todos os passageiros em simultâneo, se for, através do ciclo "*while*" (linha ❹❷), as imagens finais dos passageiros são enviadas para uma *script* chamada "imagens.sh" (linha ❹❸), para que sejam colocadas como argumentos à aplicação *showfoto* (visualizador de imagens).

Terminado o ciclo "*while*", a *script* é executada (linha ❹❹) e todas as imagens finais dos passageiros são apresentados no ecrã, ficando assim toda a informação disponível para consulta.

Se a opção escolhida for outra, significa que se quer consultar a informação apenas de um passageiro, por isso, na linha ❹❺, é chamada a imagem final que lhe corresponde.

4.5.2.8 Eliminação de Dados

No caso de haver a necessidade de apagar toda a informação relativa a um passageiro, foi criada a opção 7, que apresenta o "Menu Auxiliar - Eliminação" (Figura 3.7).

Antes da função, que apaga o perfil, ser chamada é executada a função "existeparaapagar()" (Figura 4.15) que verifica se o perfil existe ou não na base de dados do sistema. Só no caso de existir é que a função "apaga()" (Figura 4.16) é chamada.

```
existeparaapagar()
{
clear
k=$(cat perfil$1.txt)
if [ "$k" == "ESTE PASSAGEIRO NÃO ESTÁ REGISTADO" ]; then
echo "ESTE PASSAGEIRO NÃO ESTÁ REGISTADO"
sleep 2
menu
fi
}
```

Figura 4.15: Função que verifica se o Passageiro existe na Base de Dados para ser Eliminado

```
apaga ()
{
❹❻  rm -fr perfil$1.txt
❹❼  rm -fr passageiro$1.jpg
❹❽  rm -fr fotos/pessoa$1.jpg
❹❾  echo  ESTE PASSAGEIRO NÃO ESTÁ REGISTADO > perfil$1.txt
❺❿  cp fotos/fundocinza.jpg /fotos/pessoa$1.jpg
    clear
    echo "Perfil $1 Apagado"
    sleep 2
    return
}
```

Figura 4.16: Função de Eliminação de Passageiros do Sistema

A função "apaga()" começa por eliminar os dados relativos ao passageiro escolhido, o ficheiro do perfil ("perfilN.txt"), a imagem final ("passageiroN.jpg") e a foto do passageiro ("pessoaN.jpg"), linhas ❹❻, ❹❼ e ❹❽ respectivamente. A linha ❹❾ cria de novo o ficheiro de perfil com o texto que indica à aplicação do sistema que o passageiro eliminado não existe; a linha ❺❿ coloca no lugar da fotografia deste passageiro uma outra com fundo cinzento de modo a que, quando a informação é enviada do cliente para o servidor, vá na posição relativa ao passageiro eliminado uma imagem final igual à que se vê na Figura 4.17.

Figura 4.17: Imagem Final de um Passageiro que não Existe no Sistema

4.5.2.9 Envio de Dados

A opção número 8 activa o SEB de forma manual, permitindo aos ocupantes, com conhecimento do código de segurança, activar o sistema em situações em que não tenha acontecido um acidente grave, tais como diagnósticos ao sistema.

Esta opção da aplicação do sistema é um complemento ao envio automático, e é implementada pela função "envia()" (Figura 4.18).

```
envia()
{
      o=1
      echo 0 > flag.txt
      max=$(cat maxocup.txt)
      let max=$max+1
      echo -n A PROCURA DE UM MEIO DE SOCORRO


51 tar zcf media.tgz pa*.*
52 python rfcomm-client.py media.tgz &

     vflag=$(cat flag.txt)
53 while [ $vflag -eq 0 ]
"" do
""    echo -n .
""    sleep 0.5
""    vflag=$(cat flag.txt)
""    done
54 envia
```

Figura 4.18: Função que chama o módulo de Envio de Dados

Depois de iniciar variáveis, a função, na linha 51 faz a compactação das imagens finais ("passageiroN.jpg"), da matrícula ("pamatricula.txt") e da lotação ("palotacao.txt") do veículo, para que toda a informação seja enviada através de um único ficheiro, "media.tgz". Esse ficheiro é colocado como parâmetro ao módulo de *software* que permite fazer o envio da informação do lado do cliente, que em *background* vai tentar estabelecer contacto com a entidade servidor (linha 52). O módulo de *software* desenvolvido para este efeito está guardado no ficheiro "RFCOMM-CLIENT.PY".

Enquanto é feita a tentativa de estabelecimento de comunicação, é apresentado no ecrã da aplicação, o símbolo ".", para haver a noção que existe actividade no sistema. Esta funcionalidade é implementada pelo ciclo "*while*" (linha 53).

Quando o parceiro de comunicação é encontrado, o valor do ficheiro "flag.txt" é alterado para "1" e o registo de actividade pára, tendo lugar a apresentação de uma imagem com o texto (Figura 3.9) que indica o início e o fim do envio de dados entre as entidades.

No fim do envio de dados e, enquanto não for encontrado o parceiro de comunicação, a função "envia()" é chamada de novo (linha 54) e permanece em ciclo até a aplicação do sistema ser desligada. O objectivo é que a entidade cliente continue à procura de outros meios de socorro ou autoridades a quem possa enviar dados, para lhes fornecer a informação biomédica.

4.5.3 Módulo "RFCOMM-CLIENT.PY"

Este módulo foi desenvolvido para permitir a comunicação entre o cliente e o servidor, tal como foi descrito na Subsecção 4.3**.** Cabe aqui fazer a descrição da sua implementação, que se pode ver na Figura 4.19.

```
    from bluetooth import *
    import sys

    # A Procura do Serviço de Emergencia Bluetooth
    uuid = "94f39d29-7d6d-437d-973b-fba39e49d4e1"
55  service_matches = find_service( uuid = uuid )

56  if len(service_matches) == 0:
        g = open("flag.txt","w")
57      g.write("1")
        g.close()
58      print "\n\n        ---NENHUM MEIO DE SOCORRO ENCONTRADO---"
        print "                        --NOVA PROCURA--\n"
59      sys.exit(0)

60  first_match = service_matches[0]
""  port = first_match["port"]
""  name = first_match["name"]
""  host = first_match["host"]

61  print "\n\n             ---ENCONTRADO UM MEIO DE SOCORRO---\n"
""  print "                      ---A ENVIAR DADOS---"

   # Criação do socket do cliente
62  sock=BluetoothSocket( RFCOMM )
63  sock.connect((host, port))

64  f=open(sys.argv[1],"rb")
65  data=f.read()
66  sock.send(data)

67  f.close()
68  sock.close()

69  g = open("flag.txt","w")
""  g.write("1")
""  g.close()

70  print "            --ENVIO CONCLUÍDO--\n              "
""  print "        -INÍCIO DA PROCURA  DE OUTRO MEIO DE SOCORRO-\n  "
```

Figura 4.19: Módulo "RFCOMM-CLIENT.PY"

As primeiras linhas de código são destinadas ao carregamento dos módulos "bluetooth" e "sys". Através deles, é possível implementar o SDP na entidade cliente para procurar o serviço criado, também, pelo SDP na entidade servidor.

O serviço é criado na entidade servidor com dois elementos que o identificam, que são o nome e o UUID (Universally Unique Identifier). O UUID consiste num código único e complexo de 128 bits que vai garantir que apenas as entidades cliente autorizadas (com conhecimento do UUID) usem o serviço. Mesmo que alguém mal intencionado replique o código dos módulos de *software* com intenções de receber a informação biomédica, não o vai conseguir, se não possuir o UUID.

Definido o código que identifica o serviço, inicia-se a busca (linha ❺❺). O método "find_service" procura um serviço numa ou várias entidades e recebe como parâmetro o UUID. Da aplicação deste método resultam os seguintes valores: canal de comunicação; endereço do dispositivo que anuncia o serviço e nome desse serviço. Isto só é possível se estiver alguma entidade Bluetooth a anunciar o serviço ao alcance, senão os valores resultantes da aplicação do método serão nulos.

Se forem nulos (linha ❺❻), o valor do ficheiro "flag.txt" é alterado para "1" (linha ❺❼) para que o registo de actividade implementado na linha ❺❸ (Figura 4.18) pare. Logo de seguida, a aplicação do sistema apresenta no ecrã uma mensagem (linha ❺❽) a indicar que nenhum meio de socorro foi encontrado e que se inicia uma nova procura. A execução do módulo termina na linha ❺❾, e de seguida a aplicação do sistema chama de novo a função "envia()" para que esta continue à procura de meios de socorro a quem possa enviar informação.

Não sendo nulos, os valores resultantes da aplicação do método, são atribuídos às variáveis "port", "name" e "host" (linha ❻❿) e a aplicação do sistema apresenta no ecrã uma mensagem a indicar que um meio de socorro foi encontrado (linha ❻❶).

O *socket* RFCOMM da entidade cliente é criado (linha ❻❷) e com os valores obtidos inicia a comunicação com o *socket* RFCOMM da entidade servidor (linha ❻❸). A partir deste momento, a ligação está estabelecida entre as duas partes, sendo possível fazer o envio da informação da entidade cliente para a entidade servidor.

O processo de envio da informação inicia-se com a abertura do ficheiro "media.tgz" para leitura (linha ❻❹) que é colocado como único argumento ao módulo "RFCOMM-CLIENT.PY". Na linha ❻❺, o conteúdo desse ficheiro é lido para uma variável, e enviado

através do *socket* criado (linha ❻❻). Terminado o envio de dados é fechado o acesso ao ficheiro "media.tgz" (linha ❻❼), e encerrado o *socket* (linha ❻❽), terminando a comunicação entre as partes.

A linha ❻❾ é responsável por alterar o valor do ficheiro "flag.txt", para indicar ao processo, que regista a actividade do sistema, que deve terminar.

A construção deste módulo termina com a exibição da mensagem de conclusão da transmissão entre as partes (linha ❼❿) e que indica, de seguida, que terá lugar a procura por outro meio de socorro.

4.5.4 Alteração do Código de Segurança

Por defeito, a aplicação do sistema é implementada com um código de segurança, cuja possibilidade de alteração existe, através da opção número 8 do "Menu Principal", que chama a função "alteracodigo()", visível na Figura 4.20.

```
alteracodigo ()
{
❼❶    echo "Insira o Novo Código"
""    read newcode
""    echo "Por Favor Confirme o Novo Código"
""    read newcodeconf
❼❷    if [ $newcode -eq $newcodeconf ] ; then
""     echo $newcode > pin.txt
       else
         echo "Códigos Diferentes. Volte a Inserir"
         echo "Pressione qualquer tecla. . ."; read
❼❸       alteracodigo
       fi

    echo "Código Alterado com Sucesso"
    echo "Pressione qualquer tecla. . ."; read
}
```

Figura 4.20: Função que permite a alteração do Código de Segurança da Aplicação do Sistema

Esta simples função começa por pedir ao utilizador que introduza o novo código e a sua confirmação (linha ❼❶). Se forem idênticos, o novo código é escrito no ficheiro "pin.txt" (linha ❼❷), caso contrário é solicitado nova introdução (linha ❼❸).

4.6 Aplicação do Sistema, Entidade Servidor

A aplicação do sistema na entidade servidor é relativamente simples, se comparada com a entidade cliente, pois o seu "Menu Principal" (Figura 3.10) que é também desenvolvido em Linux Shell, tem apenas quatros opções.

A primeira, para activar o SEB de modo a que este procure por veículos sinistrados; a segunda, para consultar a informação biomédica já recebida e existente no sistema; a terceira, para eliminar toda essa informação quando ela não for mais precisa e, a última opção, para sair da aplicação.

Quando a aplicação é iniciada, não é pedido para introduzir qualquer código, ao contrário do que acontece na aplicação da entidade cliente. Isto porque a aplicação da entidade servidor não permite adulterar informação. Outro motivo, é a possibilidade de várias pessoas poderem utilizar os veículos de emergência e das autoridades, ou seja, se existisse um código para activar o sistema, este teria de ser conhecido por todos os utilizadores e alterado frequentemente para garantir alguma segurança, tornando o acesso à aplicação do sistema mais complexo, o que não é desejável.

4.6.1 Activação da Procura por Veículos Sinistrados

Quando no "Menu Principal", gerado pela função "menu()" (Figura 4.21) é escolhida a primeira opção, de imediato é invocada a função "recebe()" (linha ❼❹).

```
menu ()
{
 while :
 do
   clear
   echo "|=====================================================|"
   echo "|  SERVIÇO DE EMERGÊNCIA BLUETOOTH -  Menu Principal   |"
   echo "|-----------------------------------------------------|"
   echo "| [1] Activação da Procura por Veículos Sinistrados    |"
   echo "| [2] Consulta de Informação Recebida                  |"
   echo "| [3] Eliminação de Toda a Informação Recebida         |"
   echo "| [4] Sair                                             |"
   echo "|=====================================================|"
   echo -n "Escolha uma opção [1-4]>>"
   read yourch
   case $yourch in
   1) clear
 ❼❹  recebe
      ;;
   2) clear
      echo "| ============================================|"
      echo "|       Menu Auxiliar - Consulta              |"
      echo "|---------------------------------------------|"
      j=1
     exec 3< registodematriculas.txt
 ❼❺  while read <&3
      do
 ❼❻    echo "| Matrícula - $REPLY                          |"
       let j=$j+1
      done
      echo "|============================================|"
      echo -n "Insira a Matrícula do Veiculo >>"
   .
   .
   .
    esac
 done
}
```

Figura 4.21: Função que implementa o Menu da Aplicação do Sistema, Entidade Servidor

A função “recebe()” (Figura 4.22) é responsável por implementar a recepção e armazenamento dos dados enviados pela entidade cliente, isto é, as imagens finais dos passageiros, a matrícula, e a lotação de cada veículo.

Mas, para que a entidade cliente possa enviar a referida informação, é necessário que primeiro a entidade servidor crie o serviço que vai permitir estabelecer a comunicação entre as entidades. Para tal, foi desenvolvido em Phyton um módulo de *software* chamado “RFCOMM-SERVER.PY”, ao qual a função “recebe()” recorre.

```
recebe ()
{
    clear
    rm -fr imagens.sh
    rm -fr media.tgz
    rm -fr pa*
    echo 0 > bandeira.txt
    c=$(cat bandeira.txt)

    echo -n "À PROCURA DE UM VEICULO SINISTRADO"
    echo -n "/usr/bin/showfoto" > imagens.sh

 77 python rfcomm-server.py media.tgz &

 78 while [ $c -eq 0 ]
 "" do
 ""   echo -n "."
 ""   c=$(cat bandeira.txt)
 ""   sleep 0.5
 ""  done

 79 tar xf media.tgz

    d=$(cat pamaxocup.txt)
    let d=$d+1
    n=1

    f=$(cat pamatricula.txt)

 80 exec 3< registodematriculas.txt
 ""  while read <&3
 ""  do
 ""    if [ $REPLY == $f ] ; then
 ""     echo "VEICULO COM A MATRÍCULA $REPLY JÁ EXISTE NO SISTEMA"
 ""     sleep 2
 ""     recebe
 ""    fi
 ""  done
 ""  exec 3>&-

 81 mkdir -p /seb/$f
 ""  mv /seb/pa*.* /seb/$f

    echo $f >> registodematriculas.txt

 82 while [ $n -le $d ]
 "" do
 ""   echo -n " /seb/$f/passageiro$n.jpg" >> imagens.sh
 ""   let n=$n+1
 "" done

    chmod 777 imagens.sh
 83 /seb/imagens.sh &

 84 recebe
}
```

Figura 4.22: Função que chama o Módulo de Recepção de Dados e que os apresenta

O módulo "RFCOMM-SERVER.PY" é chamado na linha ❼❼ e o seu processo corre em *background* para que o ciclo, que implementa o registo de actividade do sistema, se inicie (linha ❼❽). Realizada a recepção da informação na forma do ficheiro "media.tgz", o registo de actividade pára e o ficheiro é descompactado (linha ❼❾), ficando toda a informação nele contida disponível para processamento.

Para evitar que no mesmo acidente de viação a aplicação do sistema da viatura de emergência receba mais do que uma vez a mesma informação, foi implementado o ciclo "*while*" presente na linha ❽❿. Este ciclo volta a ler o ficheiro que guarda o registo das matrículas existentes e compara-as com a matrícula que é recebida. Se a matrícula recebida já existir no sistema, os dados recebidos são descartados e é iniciada nova procura. Caso contrário, a informação recebida é armazenada (linha ❽❶).

De seguida, é adicionado à *script* "imagens.sh" o caminho para os ficheiros que contém as fotografias dos passageiros (linha ❽❷), para serem colocadas como parâmetro à aplicação *showfoto*, que as vai apresentar no ecrã da aplicação, através da linha ❽❸.

Para terminar, na linha ❽❹, é chamada de novo a função "recebe()" para que, desta forma, a aplicação do sistema continue sempre a procurar por veículos sinistrados, até que os utilizadores da aplicação na entidade servidor decidam que não desejam receber mais informação.

4.6.2 Módulo "RFCOMM-SERVER.PY"

O Módulo "RFCOMM-SERVER.PY" (Figura 4.23) foi desenvolvido em Phyton, e é responsável por criar o serviço que vai possibilitar a comunicação entre as entidades cliente e servidor.

```
   from blueooth import *
   import sys

❽❺ server_sock=BluetoothSocket( RFCOMM )
❽❻ server_sock.bind(("",PORT_ANY))
❽❼ server_sock.listen(1)

❽❽ uuid = "94f39d29-7d6d-437d-973b-fba39e49d4e1"

❽❾ advertise_service( server_sock, "SEB",service_id = uuid )

❾❿ client_sock, client_info = server_sock.accept()
    try:
        print "\n>>ENTRADA DE DADOS"
❾❶       while True:
            data = client_sock.recv(1024)
            f=open(sys.argv[1],'a')
❾❷          f.write(data)
    except IOError:
    pass

    ban = open("bandeira.txt","w")
    ban.write("1")
    ban.close()

❾❸ client_sock.close()
""  server_sock.close()
```

Figura 4.23: Módulo "RFCOMM-SERVER.PY"

No início, são importados os módulos necessários "bluetooth" e "sys".

De seguida, na linha ❽❺, é criado o *socket* RFCOMM da entidade servidor. Para que este *socket* aceite ligações, tem de ser associado a um recurso de *hardware* disponível, por exemplo um adaptador Bluetooth, e também a um canal de comunicação (linha ❽❻). Assim que é feita a associação, pode-se colocar o *socket* em modo de "escuta" ficando disponível para aceitar conexões (linha ❽❼). Na linha ❽❽, é definido qual vai ser o UUID utilizado na criação do serviço, que é criado e anunciado com o nome "SEB" (linha ❽❾).

Para a sua criação e publicitação, é essencial que haja: um *socket* criado e associado a um recurso de *hardware*, e à escuta; um nome para o serviço, e um UUID. O serviço é anunciado continuamente até ser encontrado um cliente.

Com o *socket* e o serviço criados, é possível estabelecer a comunicação com os clientes, para se começar a receber dados. Quando é encontrado um cliente a solicitar um serviço com as características do que é anunciado pelo servidor, a conexão é estabelecida (linha ❾❿).

Os dados que o cliente envia para o servidor (o ficheiro "media.tgz") são recebidos em pacotes de 1024 *bytes* durante as iterações do ciclo "*while*" (linha ❾❶), e são gravados na entidade servidor para um ficheiro com o mesmo nome (linha ❾❷). Este ficheiro, quando criado, tem de ser em modo "*append*", isto para que os pacotes recebidos nas várias iterações do ciclo não se sobreponham aos dados já recebidos.

Por fim, os *sockets* são fechados para terminarem a ligação (linha ❾❸).

4.6.3 Consulta de Informação

A opção número dois do "Menu Principal" permite fazer o armazenamento da informação biomédica que é recebida de um ou mais veículos. Quando esta opção é escolhida, é criado um menu dinâmico, a partir da linha ❼❺. Esta linha implementa um ciclo "*while*" que lê, linha à linha, o ficheiro "registodematriculas.txt" que, tal como o nome indica, possui a informação sobre as matrículas de todos os veículos dos quais a aplicação do sistema já recebeu informação biomédica. De seguida as matrículas são apresentadas como opções no "Menu Auxiliar - Consulta" (linha ❼❻). O utilizador da aplicação, ao introduzir o valor de qualquer uma das matrículas apresentadas, vai obter informação biomédica dos passageiros da viatura a que ela se refere.

4.6.4 Eliminação de Informação

Esta opção quando escolhida, elimina do sistema, todos os directórios onde são guardados os dados biomédicos relativos aos passageiros de cada veículo.

Capítulo 5

Validação e Testes ao Sistema

5.1 Validação do Sistema em Laboratório

5.2 Validação do Sistema, na Simulação de um Cenário Real

5 VALIDAÇÃO E TESTES AO SISTEMA

Este capítulo tem como objectivo testar o SEB e analisar o seu funcionamento em determinadas situações.

5.1 Validação do Sistema em Laboratório

Foram idealizados dois cenários para realizar a validação do sistema em laboratório. O primeiro, tem como objectivo demonstrar o funcionamento do sistema entre duas entidades, uma cliente, outra servidor, ou seja, entre um veículo sinistrado e um meio de socorro que se dirige ao encontro do primeiro para prestar assistência aos seus ocupantes.

No segundo cenário, pretende-se verificar o comportamento do sistema quando três meios de socorro chegam ao local de um sinistro para prestar assistência às vítimas que se encontram em duas viaturas acidentadas.

Apesar de os cenários terem sido implementados, recorrendo a computadores, achou-se melhor colocar veículos nas figuras que os representam, para que a sua representação fique mais próxima da realidade, de forma a tornar mais claro o seu entendimento.

Todos os veículos citados possuem o SEB implementado.

5.1.1 Um Veículo Sinistrado, Um Meio de Socorro

Esta é a situação mais simples em que o SEB pode operar, isto é, a existência de apenas duas entidades; uma para enviar (cliente), e outra para receber (servidor) informação.

Este tipo de cenário pode ocorrer, por exemplo, nos casos em que uma viatura e os seus ocupantes sofrem um acidente grave provocado por aluimento do pavimento; deslizamento para uma falésia; despiste por existência de óleo no pavimento, entre muitas outras hipóteses possíveis.

O cenário foi implementado com dois computadores equipados com adaptadores Bluetooth classe 1. Cada computador é representado nas figuras por uma ambulância e por um veículo sinistrado, respectivamente.

O que acontecerá neste tipo de cenário será o seguinte:

❶ Acontece um acidente grave, o SEB (na entidade cliente) é accionado (Figura 5.1), fazendo com que este procure, num ângulo de 360º, por veículos de emergência médica (entidade servidor) com o SEB activo.

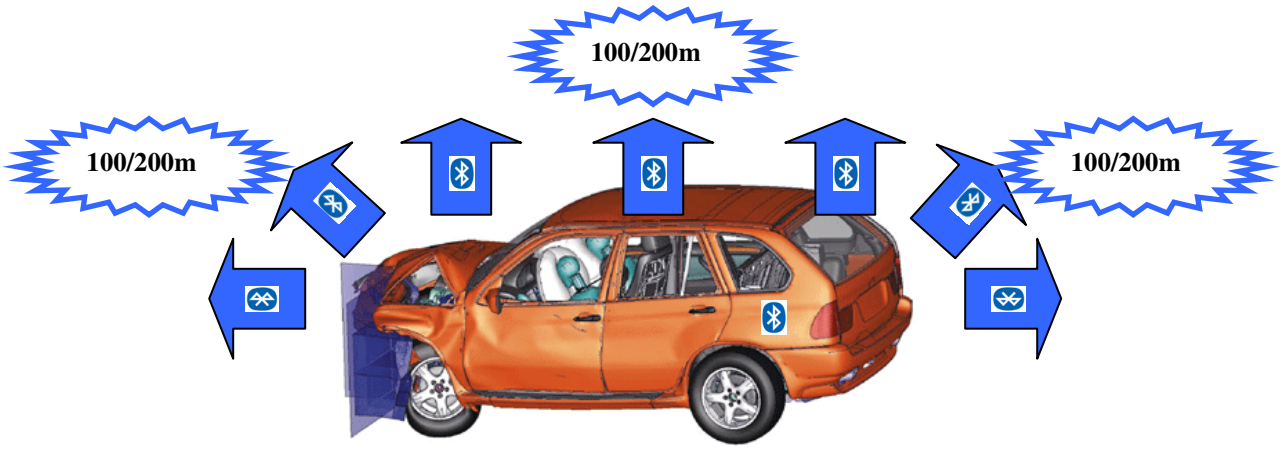

Figura 5.1: Veículo Acidentado, Entidade Cliente

❷ Alguém se apercebe do acidente e chama os serviços de emergência médica. A Figura 5.2, ilustra esse momento.

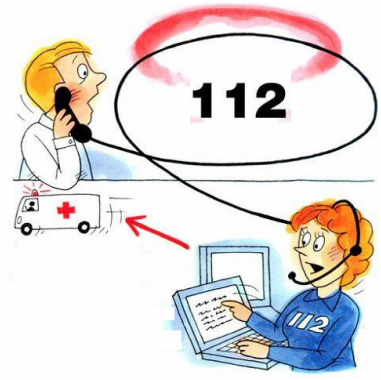

Figura 5.2: Chamada para os Serviços Centrais de Emergência Médica

❸ No momento em que os serviços de emergência médica se dirigirem para o local do acidente podem optar por activar, imediatamente, o SEB ou, então, activá-lo apenas próximo do local do sinistro.

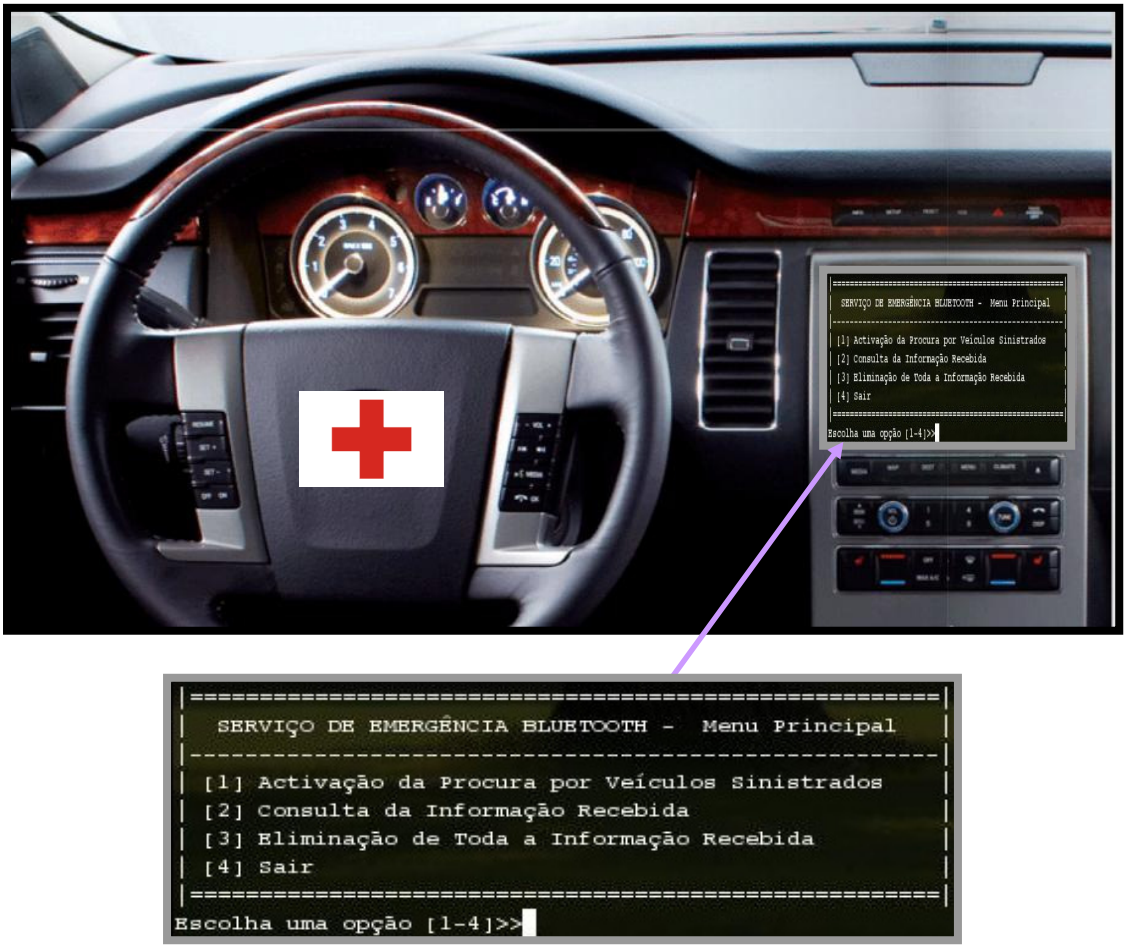

Figura 5.3: Aplicação do Sistema, Entidade Servidor, embutida na Consola do Veículo de Emergência

Seleccionando a opção 1 da aplicação do sistema (Figura 5.3), a consola passará a apresentar a imagem da Figura 5.4. É uma imagem dinâmica que faz o registo de actividade, que será continuamente apresentado até ser encontrado um ou mais veículos sinistrados.

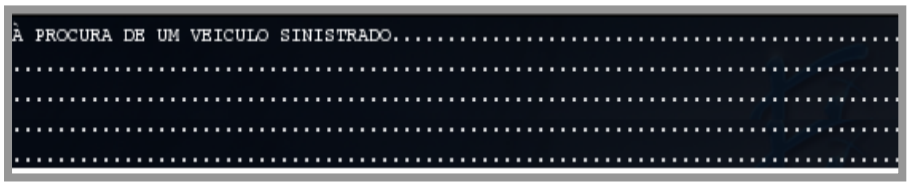

Figura 5.4: Procura por Veículos Sinistrados

❹ Quando os veículos se encontram ao alcance um do outro, ou seja, a uma distância entre si de 100 a 200 metros (alcance do sinal Bluetooth) [10], tem lugar a seguinte troca de informação: a viatura de emergência anuncia em todas as direcções, inclusive aquela onde se encontra o veículo sinistrado (Figura 5.5, seta ❶), o serviço criado pelo módulo "RFCOMM-SERVER.PY"; em resposta o SEB do veículo sinistrado envia (Figura 5.5, seta ❷), a informação biomédica disponível na sua base de dados relativa aos passageiros que poderão estar na viatura acidentada.

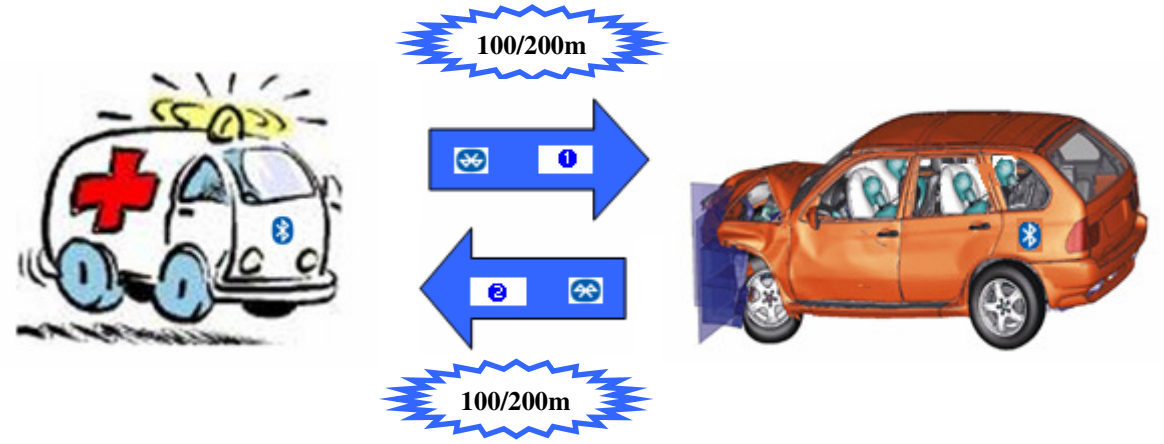

Figura 5.5: Cenário de Socorro nº 1

❺ A informação biomédica é recebida, passados alguns segundos, na viatura de emergência e apresentada na consola da aplicação do sistema (Figura 5.6).

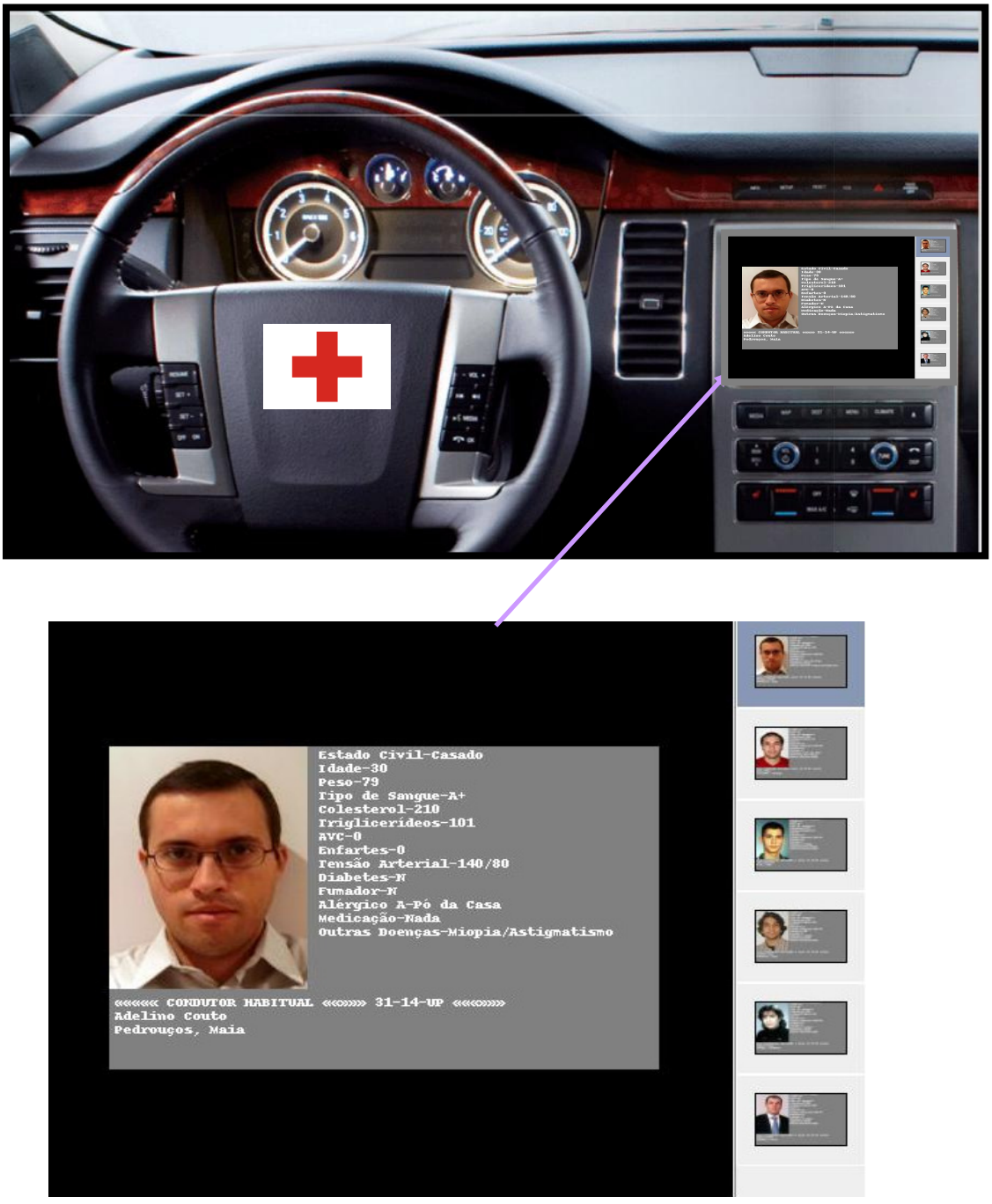

Figura 5.6: Apresentação da Informação Recebida na Viatura de Emergência Médica

Desta forma, é possível ver todos os passageiros que poderão estar no veículo, uma vez que eles aparecem todos na coluna da direita. Cabe aos profissionais de saúde pesquisar e interpretar os dados relativos a cada um.

Foram apresentados os passos em que o SEB intervém, desde o momento do acidente até ao momento que a informação biomédica chega à ambulância. Os passos seguintes serão do âmbito das competências dos profissionais de saúde.

5.1.2 Dois Veículos Sinistrados, Três Meios de Socorro

O cenário da Figura 5.7 pretende demonstrar o comportamento do serviço de emergência perante uma situação mais complexa. Este cenário foi implementado através de cinco computadores, equipados com adaptadores Bluetooth classe 1, e representados na figura por veículos.

Este tipo de situação pode ocorrer resultante de um choque frontal entre dois veículos ou de um choque em cadeia. Apesar de neste cenário estarem, apenas, representados dois veículos, envolvidos no acidente, o comportamento do sistema seria o mesmo para um número maior.

Neste cenário, os mesmos passos do teste anterior são apresentados: (passo ❶) - momento da colisão; (passo ❷) - alerta para os serviços de emergência médica, contudo, por ser um acidente de maior gravidade, são enviados mais meios; (passo ❸) - o SEB é activado nos meios de socorro; (passo ❹) - quando estes veículos se encontram ao alcance das viaturas acidentadas tem lugar a troca de informação. As diferenças substanciais existem no (passo ❺), pois cada meio de socorro receberá a informação biomédica dos passageiros de cada veículo acidentado e não a deve receber duplicada, isto é, receber a informação do mesmo veículos duas ou mais vezes.

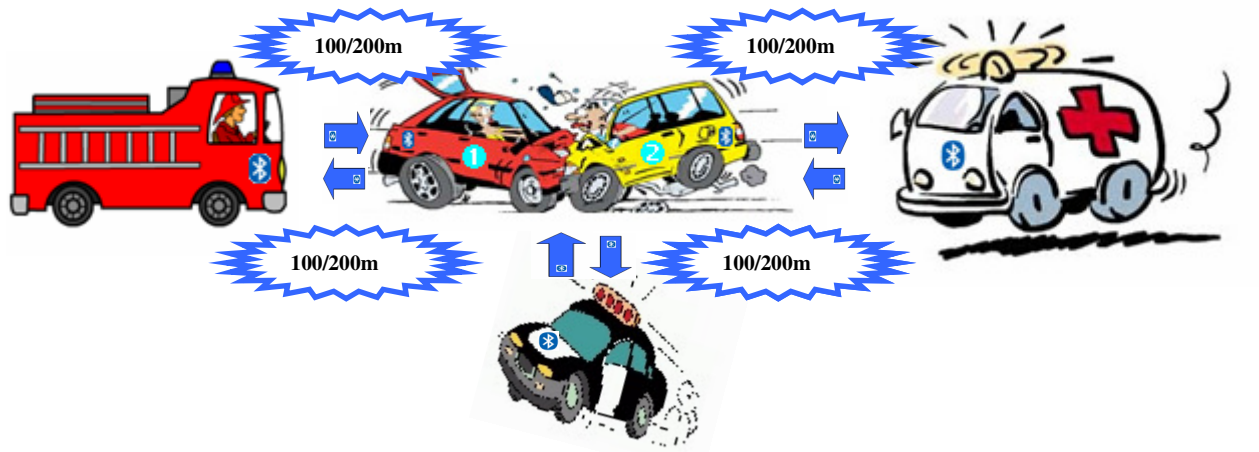

Figura 5.7: Cenário de Socorro nº 2

Cada meio de socorro recebe, integrado na consola da sua aplicação, conforme a ordem de chegada ao local do sinistro, as imagens presentes nas Figuras 5.8 e 5.9 em tempos distintos, respeitantes aos condutores habituais dos veículos envolvidos no acidente, já que neste cenário eram os únicos passageiros registados na base de dados do SEB dos veículos.

Figura 5.8: Condutor Habitual, Veículo ❶

Figura 5.9: Condutor Habitual, Veículo ❷

A melhor forma de distinguir os ocupantes é pela matrícula da viatura, que se encontra à direita da designação do tipo de passageiro.

Os cenários apresentados indicam apenas algumas das situações onde o SEB pode actuar, sendo este, útil em muitas outras.

5.1.3 Medição Estática dos Tempos de Comunicação

As medições foram realizadas tendo por base o cenário apresentado na Subsecção 5.1.1, ou seja, com duas entidades distintas. Para as realizar, foram utilizados dois computadores portáteis, os dois equipados com adaptadores Bluetooth Conceptronics classe 1, idênticos ao da Figura 5.10.

Figura 5.10: Adaptador Bluetooth Conceptronics Classe 1

As medições foram efectuadas de forma estática e em três distâncias distintas.

Os histogramas presentes nas Figuras 5.11, 5.12 e 5.13, mostram os tempos de comunicação observados às distâncias de 1, 10 e 100 metros, respectivamente, após a realização de 20 amostragens. Para estas distâncias, a probabilidade dos tempos de comunicação estarem abaixo dos 8 segundos é de, respectivamente, 75%, 55% e 75%.

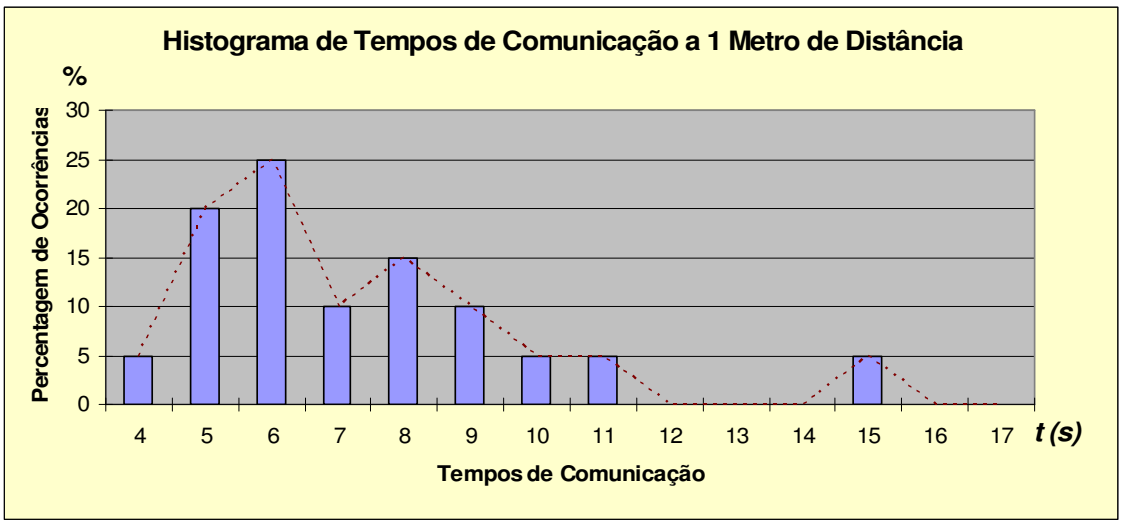

Figura 5.11: Histograma dos Tempos de Comunicação, relativos a 1 Metro de Distância

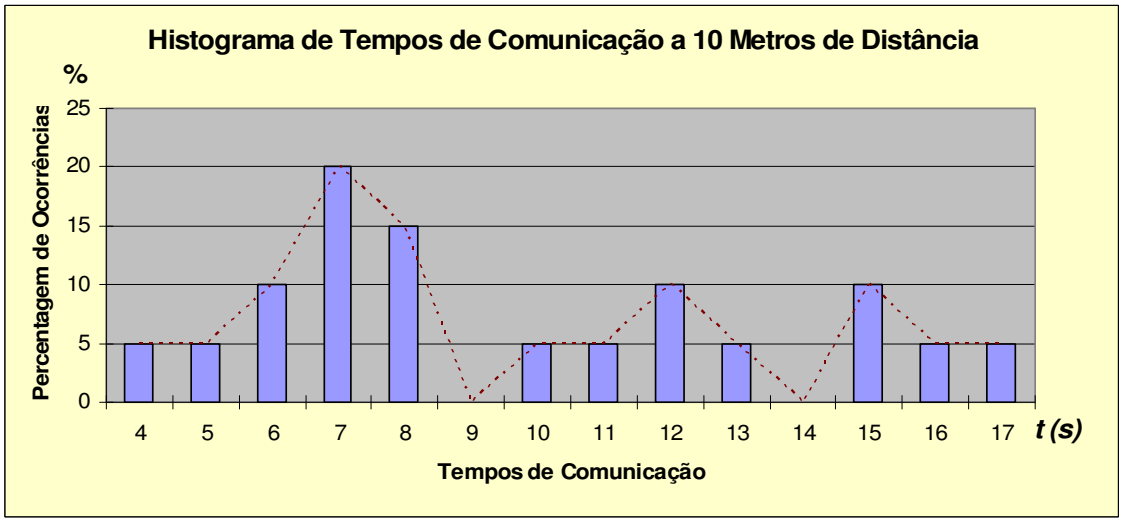

Figura 5.12: Histograma dos Tempos de Comunicação, relativos a 10 Metros de Distância

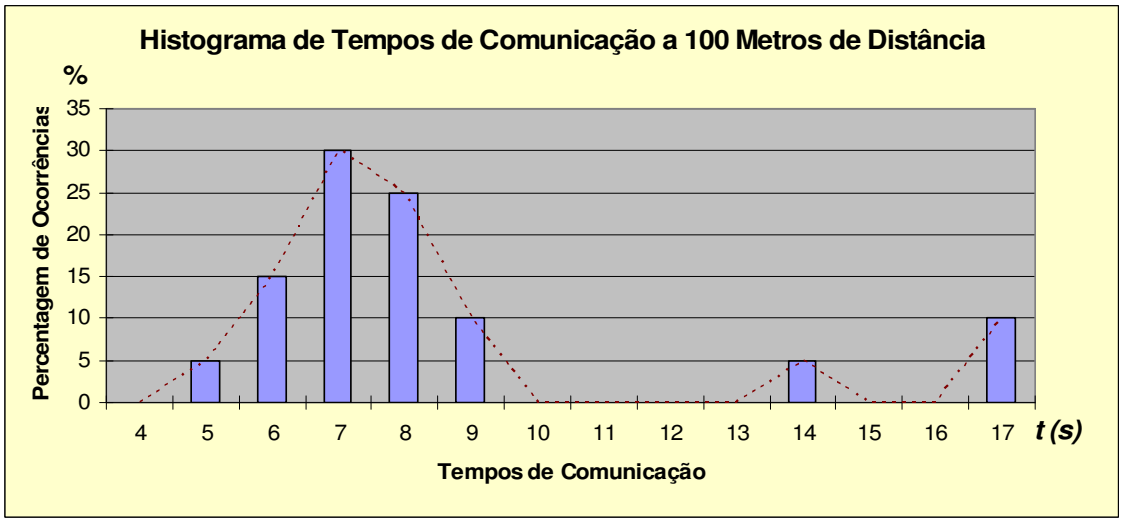

Figura 5.13: Histograma dos Tempos de Comunicação, relativos a 100 Metros de Distância

5.2 Validação do Sistema, na Simulação de um Cenário Real

Os testes realizados na simulação de um cenário real visam validar o funcionamento do sistema num ambiente semelhante ao que haverá numa situação de socorro. A situação escolhida foi a da Figura 5.5 que pertence ao primeiro cenário apresentado nos testes em laboratório.

Na simulação de um cenário real, a entidade servidor, ou seja, o meio de socorro, foi implementado por um automóvel, que transportou um computador portátil, equipado com um adaptador Bluetooth classe 1 colocado no tejadilho (Figura 5.14).

Figura 5.14: Adaptador Bluetooth no tejadilho do Automóvel

O veículo sinistrado, ou seja, a entidade cliente foi implementado, por um computador portátil, equipado também com um adaptador Bluetooth classe 1, que permaneceu sempre no mesmo local, e que representou o sítio onde ocorreu o acidente.

Ambos os adaptadores Bluetooth, durante a simulação e medições, foram colocados de forma a minimizar o efeito de interferências na potência do sinal, evitando obstáculos, tais como árvores, edifícios, postes, outros veículos, ou seja, encontravam-se em campo aberto e em linha de vista.

5.2.1 Distância Máxima de Envio e Recepção

O sinal dos dispositivos Bluetooth classe 1 tem um alcance entre os 100 e os 200 metros, valor que varia dependendo da existência ou não de obstáculos entre eles. Por este motivo, é importante medir com mais precisão, no cenário descrito, o raio de alcance do sinal Bluetooth, ou seja, a distância máxima a partir da qual as entidades cliente e servidor começam a comunicar.

Esta informação é importante para efectuar as medições dinâmicas dos tempos de comunicação entre as duas entidades, pois é a partir desta distância que é iniciada a contagem do tempo que medeia a entrada do veículo de socorro no raio de alcance do sinal Bluetooth, e a recepção da informação biomédica dos passageiros do veículo sinistrado.

A distância máxima foi obtida da seguinte forma: os testes em laboratório revelam que a 100 metros o sistema funciona eficazmente. Assim, a partir desta distância, continuaram-se a realizar medições, em intervalos de 10 metros, até que se chegou ao valor final de 250 metros. A esta distância foram realizadas outras vinte amostragens para ser feita uma comparação com as amostragens já realizadas a 1, 10 e 100 metros.

Os resultados apresentados na Figura 5.15 são semelhantes aos apresentados nas Figuras 5.11, 5.12 e 5.13, sendo a percentagem de ocorrências abaixo dos 8 segundos de aproximadamente 60%.

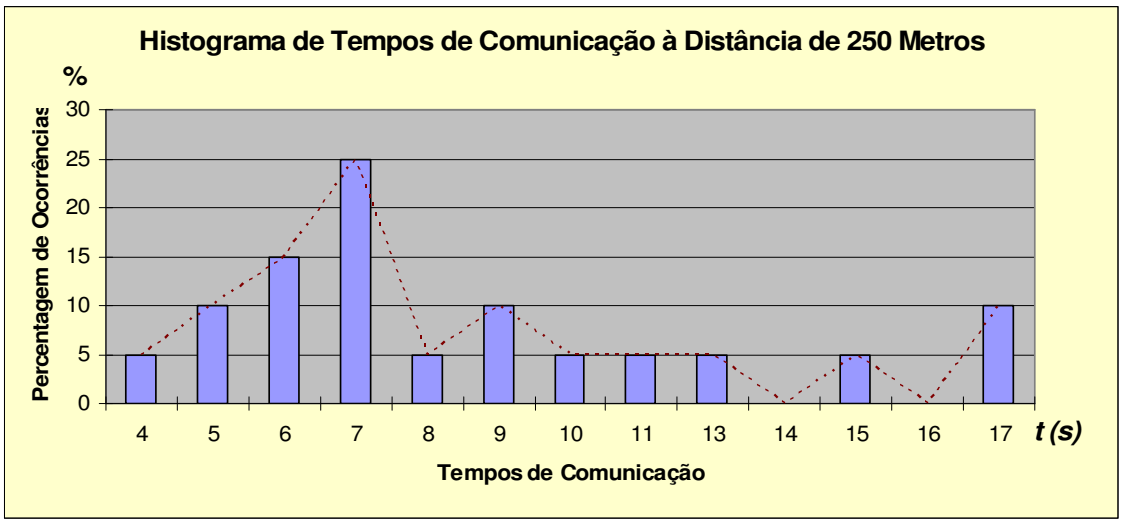

Figura 5.15: Histograma dos Tempos de Comunicação à Distância Máxima Estimada

5.2.2 Medição Dinâmica dos Tempos de Comunicação

Os tempos foram medidos de forma dinâmica, ou seja, com o veículo, que representa o meio de socorro em movimento, a uma velocidade de cerca de 45 km/h, vindo a uma distância superior a 250 metros.

A partir do momento em que o veículo em movimento entra no perímetro referido é iniciado um temporizador, que só vai ser parado no momento em que a informação biomédica dos passageiros do veículo sinistrado é apresentada, na consola da aplicação residente no meio de socorro. O histograma das medições realizadas pode ser visto na Figura 5.16.

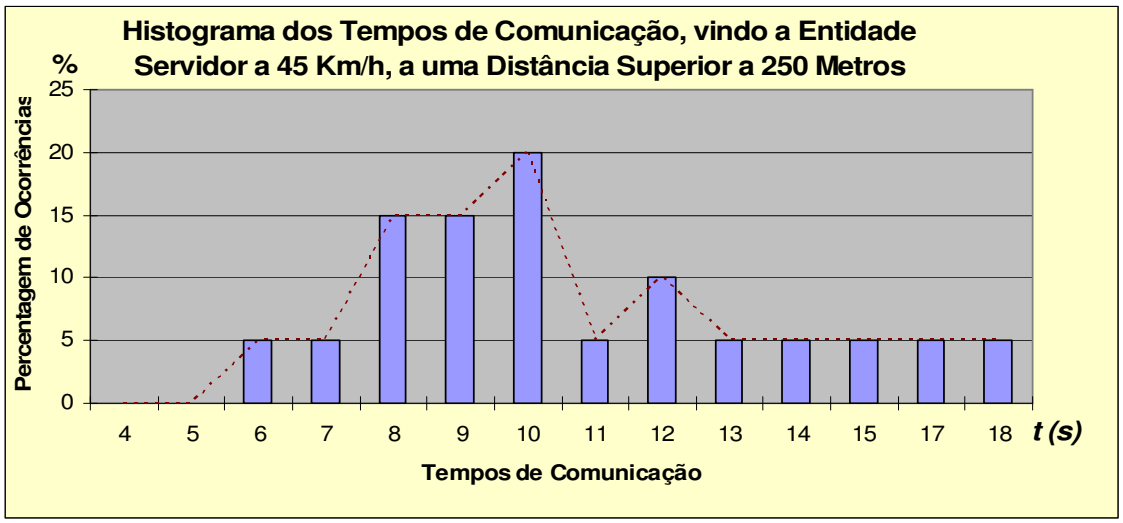

Figura 5.16: Histograma dos Tempos de Comunicação, Entidade Servidor em Aproximação

Foram tiradas vinte amostras que revelaram que o sistema continua a responder com uma percentagem de 60%, agora abaixo dos 10 segundos.

Um veículo automóvel a 45 km/h, percorre 12,5 m/s, demorando 20s desde a distância máxima estimada do alcance do SEB, até ao local do sinistro, o que pelas medições feitas, significa que há uma probabilidade de praticamente 100% do veículo de emergência receber os dados biomédicos dos ocupantes do veículo sinistrado antes de chegar ao local do sinistro, isto porque o tempo máximo de recepção foi de 18,01s.

Não é de estranhar que o tempo de resposta tenha subido ligeiramente, relativamente às medições em laboratório, uma vez que elas foram realizadas com um veículo em movimento, o que implica uma maior probabilidade da existência de interferências, com o sinal Bluetooth das duas entidades.

Capítulo 6

Conclusões

6.1 Resultados

6.2 Objectivos e Ideias para Trabalhos Futuros

6 CONCLUSÕES

Quando este trabalho começou a dar os primeiros passos não havia a certeza absoluta que o caminho traçado seria o melhor. Contudo, hoje com os resultados obtidos, estamos seguros que o caminho percorrido foi o correcto.

O problema inicial consistia em criar um novo serviço, baseado na tecnologia de comunicação sem fios Bluetooth, orientado para a sociedade civil.

Com o amadurecer das ideias tornou-se claro que um dos problemas que poderia ser abordado por esta nova tecnologia, seria o problema da sinistralidade automóvel e consequente acção de emergência médica.

Nesse sentido, foi desenvolvido um sistema baseado no paradigma cliente-servidor cuja implementação se destina a todos os veículos automóveis, nomeadamente, aos veículos civis e aos veículos de emergência médica e autoridades. Este sistema foi a base da criação de um novo serviço chamado SEB que permite aos serviços de emergência médica e autoridades, receberem a informação biomédica relativa aos ocupantes das viaturas sinistradas ao chegarem ao local de um acidente de viação.

A solução encontrada para a construção do sistema foi fazer a união e a adaptação das tecnologias Linux, Python e Bluetooth, conjuntamente com o desenvolvimento (programação) de uma aplicação cliente-servidor, implementando o conceito desenvolvido.

6.1 Resultados

Neste momento, o sistema encontra-se a funcionar de forma capaz, o que significa que as entidades cliente e servidor conseguem comunicar e trocar informação entre si de forma fiável.

Os testes em laboratório e na simulação de um cenário real, revelaram uma boa capacidade de comunicação ao nível do tempo de resposta e do alcance do SEB. Sendo que os tempos de envio e recepção de dados variam entre os 3,99s até aos 17,39s, nos testes em laboratório, e dos 6,29s até aos 18,01s, nos testes da simulação de um cenário real. A nível da distância de funcionamento do SEB, foi registado um alcance máximo de 250 metros, que pelos testes efectuados é suficiente para que os veículos de emergência médica recebam a informação biomédica dos passageiros dos veículos sinistrados, em tempo útil.

6.2 Objectivos e Ideias para Trabalhos Futuros

O sistema tem ainda uma larga margem para evoluir, especialmente no que respeita à sua integração com os veículos automóveis. É necessário desenvolver um protocolo que permita ao SEB comunicar com os sistemas de segurança que existem, actualmente, na maioria dos veículos, para que o serviço seja activo apenas nos casos em que o acidente é considerado grave. Um exemplo disso é a abertura dos *airbags* que, normalmente, só ocorre nessas situações. Contudo, para que esse desenvolvimento ocorra, é necessário o diálogo com representantes da indústria automóvel, pois, como é óbvio, terá de ser um trabalho conjunto.

O SEB para ficar integrado no mundo real, em funcionamento com os sistemas de comunicação já existentes, precisa que exista cooperação entre três entidades: a que desenvolveu o serviço, a indústria automóvel e as entidades governamentais de emergência que operam em cada país, no caso de Portugal, o INEM. Para que tal aconteça, caberá à entidade que desenvolveu o serviço efectuar diligências perante as outras, para que estas reconheçam a utilidade do serviço e o possam desenvolver em parceria.

Existe também a possibilidade futura da integração do serviço com os sistemas de informação que existam, ou que venham a existir, nas instituições hospitalares, para que a informação recolhida pelas viaturas de emergência médica, no local dos sinistros, possa ser transferida ao chegar às unidades hospitalares para esses sistemas, a fim de, por exemplo preencher a ficha de entrada da vítima.

Apesar da utilização do serviço criado ser orientada para os serviços de emergência médica, ele pode ser adaptado para receber novas funções, como por exemplo, a troca de dados pessoais entre intervenientes de acidentes de menor gravidade, já que, muitas das vezes, o tempo gasto para procurar documentos e tomar nota dos dados tem um grande e nocivo impacto sobre o tráfego automóvel, provocando quase sempre vastos congestionamentos. A existência de um serviço que possibilite a troca rápida e segura dessa informação será de grande valia para todas a partes afectadas.

Também no âmbito do controlo realizado pelas entidades que regulam o trânsito em Portugal, a GNR e a PSP, o serviço pode ser bastante útil. Os agentes da autoridade para controlar/inspeccionar um veículo apenas teriam de solicitar ao dono do mesmo a activação do serviço. Assim, os agentes receberiam na sua viatura todos os dados relativos ao condutor e ao seu veículo. Os dados a receber teriam de ser definidos na altura do desenvolvimento do serviço.

Outra solução, que poderá ser implementada no futuro, será a automatização da inserção da informação na aplicação do sistema. Neste momento, a inserção é feita manualmente, mas com a crescente utilização das tecnologias Bluetooth (1,5 mil milhões de dispositivos fabricados desde 1998 a 2008 [3]), cada vez mais estamos rodeados por equipamentos deste género. Fazendo com que a possibilidade, de no futuro, quando alguém com um dispositivo Bluetooth (género cartão de crédito, com *chip* Bluetooth), que contenha a sua informação biomédica, entrar num veículo equipado com o SEB, essa informação será transferida para o SEB do automóvel. Da mesma forma, no momento em que os passageiros acabam a sua viagem, ao saírem da viatura, a informação relativa a cada um deles é retirada do sistema.

Assim estabelecendo objectivos e ideias para trabalhos futuros termino a escrita da minha dissertação.

Ela permitiu obter um conhecimento mais alargado sobre o Bluetooth e tecnologias associadas, possibilitando o desenvolvimento de algo novo que pode ser utilizado para o bem-estar da sociedade civil.

O meu desejo é que, em breve, o SEB, com a colaboração das entidades referidas anteriormente, possa "saltar" do papel desta dissertação para o mundo real, cumprindo o objectivo para o qual foi desenvolvido.

Bibliografia

[1] Albert S. Huang, Larry Rudolph. *Bluetooth Essentials for Programmers*. Paperback 2007. 210p. ISBN-13: 9780521703758

[2] ANACOM. SIRESP. *Comunicações de Emergência e Segurança*. http://www.icp.pt/template16.jsp?categoryId=145722

[3] Bluetooth SIG. Signature Magazine. *The Beat Goes On.* http://www.nxtbook.com/nxtbooks/bluetooth/signatureq108/

[4] BlueZ Project. *BlueZ - Official Linux Bluetooth protocol stack*. http://www.bluez.org

[5] Eric Foster-Johnson, John C. Welch, and Micah Anderson. *Beginning shell scripting*. Indianapolis:Wiley Pub., 2005. 510p. ISBN 0764583204

[6] Gast, Matthew. *802.11 wireless networks : the definitive guide*. 2ª ed. Sebastopol: O'Reilly, 2005. 630p. ISBN 0596100523

[7] Kaaranen, Heikki. *UMTS networks : architecture, mobility and services*. 2ª ed. Chichester: John Wiley & Sons, 2005. 406p. ISBN 0-470-01103-3

[8] Mark Lutz, David Ascher. *Learning Python*. 2ª ed. Beijing: O' Reilly, 2004. 591p. ISBN 978-0-596-00281-7

[9] Ministério da Saúde. INEM. *Missão*. http://www.inem.pt/pagegen.asp?SYS_PAGE_ID=468559

[10] Muller, Nathan J. *Bluetooth Demystified*. New York: McGraw Hill, 2001. 396 p. ISBN 0071363238

[11] Siegmund M. Redl, Matthias K. Weber, Malcom W. Oliphant. *An introduction to GSM*. Boston: Artech, 1995. 379p. ISBN 0-89006-785-6