“A pessoa que usa as ferramentas
de ontem em trabalhos de hoje
não estará nos negócios amanhã.”

Business Week, the McGraw-Hill Companies

Agradecimentos

Pretendo agradecer a todos aqueles que, nos últimos dois anos, de alguma forma contribuíram para a elaboração desta dissertação e para a minha formação académica, profissional e pessoal.

Em primeiro lugar, quero salientar o meu apreço e agradecimento ao Professor Doutor José Francisco Morgado, pela forma como me orientou e pelo seu contributo, que foi imprescindível para o resultado final que é apresentado neste documento.

Sem diferenciar nomes, com receio de me esquecer de alguém importante, agradeço aos meus colegas, amigos e professores.

Por último, mas de forma alguma menos importantes, à minha família e à minha namorada por toda paciência e incentivo transmitido durante o desenvolvimento desta dissertação

A todos eles, sem qualquer distinção: Muito obrigada!

Resumo

A atual crise económica de abrandamento e recessão leva a que várias empresas estejam a sentir elevadas quebras no seu volume de negócios e de produção. Deste modo a inovação e a diferenciação assumem um papel de relevo nas empresas especialmente em tempos de crise.

Tradicionalmente, as empresas especializadas em reparações de parapentes utilizam meios meramente visuais para avaliar o estado do cone de suspensão dos parapentes[1]. Este tipo de inspeções é pouco fiável e coloca em causa a segurança do piloto.

Embora o parapente seja considerado um desporto seguro, existem alguns riscos associados a esta prática desportiva. O estado do material utilizado assume um papel preponderante no que diz respeito à segurança. Dependendo da sua composição, os cordões apresentam diferentes características e comportamentos: enquanto uns apresentam maior estabilidade a nível dimensional e menor flexibilidade, outros apresentam menor estabilidade dimensional e maior flexibilidade. Contudo, com o tempo e os esforços a que são sujeitos, todos vão sofrendo desgaste, provocando variações ao nível da sua dimensão e perdendo resistência. [25]

É, neste contexto, que aqui é proposto um novo método de avaliação do cone de suspensão. Para isso, foi desenvolvido um protótipo, designado por *sistema de testes para cone de suspensão de parapente* que permite analisar uma amostra de cordões e desta forma avaliar o seu estado atual. Este tipo de testes possibilita ao utilizador aferir do estado da asa e daí decidir qual a melhor forma de reparar/aumentar a segurança da asa.

[1] Conjunto de linhas que suportam o peso do piloto

Abstract

The current economic slowdown conjecture leads to recession and many companies are experiencing high losses in its turnover and production. This way, the innovation and differentiation are an important role in business especially in times of crisis.

Traditionally, companies that specialize in repairs of paragliders using only visual means to assess the state of the cone suspension of paragliders[2]. Such inspections are unreliable and jeopardize the safety of the pilot.

Although the glider is considered a safe sport, there are some risks associated with this sport. The state of the material takes a leading role with regard to safety. Depending on its composition, the strings have different characteristics and behaviors: while some have more stable and less dimensional flexibility, others are less dimensional stability and flexibility. However, with time and effort that they are subject, everyone will suffer wear and tear, including changes in terms of size and losing strength. [25]

It is in this context that we propose here a new method for evaluating the cone suspension. For this, we developed a prototype, called the testing system for rear suspension that allows the glider to analyze a sample of cord and thus assess its current state. This type of testing allows the user to gauge the state of the wing and then decide the best way to fix / enhance the security of the wing

[2] Set of lines that support the weight of the pilot

Índice

Índice de figuras

Lista de Tabelas

Lista de acrónimos

BLL	*Business Logical Layer*
CDM	*Conceptual Data Model*
DAL	*Data Acess Layer*
DFD	*Diagrama de fluxo de dados*
ER	*entity-relationship*
GPL	*General Public License*
PC	*Personal Computer*
PDM	*Physical Data Model*
SCADA	*Supervisory Control and Data Aquisition*
SGBD	*Sistema de Gestão de Base de Dados*
SQL	*Structured Query Language*
UI	*User Interface*
UML	*Unified Modeling Image*

Capítulo 1

Introdução

Neste capítulo introdutório faz-se o enquadramento do trabalho, assim como se explicita a motivação que esteve na sua origem e os objetivos que foram inicialmente estabelecidos.

1.1 Enquadramento, motivação e objetivos

A prática do parapente tem crescido exponencialmente durante os últimos anos. Trata-se de um desporto em que os seus utilizadores atingem altitudes muito elevadas. Desta forma, é indispensável garantir que toda a estrutura do parapente se encontra em bom estado.

No que diz respeito à verificação do estado do tecido, as empresas especializadas em reparações de parapente já efetuam alguns testes que possibilitam averiguar quanto ao seu estado. No entanto, no que diz respeito aos cordões, são realizados testes meramente visuais, que não são precisos e que podem por o piloto em risco. Poder-se-á dizer que o praticante…. "Arrisca a sua vida por um fio!"

Sendo este um desporto de competição, o capítulo da performance é tido como de elevada importância. Neste sentido, a espessura dos fios[3] que constituem o parapente tem vindo a diminuir consideravelmente. Não obstante a tecnologia de produção de cordas/fios ter evoluído bastante nos últimos anos, a diminuição de espessura dos fios aumenta ainda mais o nível de preocupação com as questões de segurança.

Nos últimos anos, têm-se assistido ao surgimento de uma nova modalidade, "parapente com motor", designada atualmente como paramotor. Esta, permite ao parapente uma nova versatilidade, no entanto, o peso adicional faz com que a

[3] Ao longo da dissertação o substantivo linhas tem o mesmo significado que cordões

estrutura geral fique mais pesada e desta forma a pressão exercida sobre os fios seja maior [28].

A inexistência de testes específicos para cordões de parapente nas empresas especializadas, faz com que a ideia de criar um sistema de testes que permita averiguar do real estado deste tipo de material ganhe sentido. Assim nasce a ideia de criar um sistema de testes para cone de suspensão de parapente

.O principal objetivo a atingir com a realização deste trabalho consiste no desenvolvimento de um protótipo que permita obter, de forma precisa, diversas conclusões quanto ao estado dos cordões que constituem o cone suspensão de um parapente. O sistema deverá ser capaz de efetuar vários cálculos (desvio médio, estado do cone de suspensão), assim como efetuar representações gráficas, comparando os valores obtidos no teste a determinado cordão, com os seus valores de fábrica. Os resultados obtidos deverão ser utilizados, não apenas para análise imediata do estado de um cordão, mas também para deduzir tipos de comportamentos dos diversos tipos de cordões tornando o grau de fiabilidade de reparação mais elevado.

1.2 Estrutura da Dissertação

No primeiro capítulo é contextualizado o tema da dissertação, ou seja, o problema a tratar. Identificam-se também os objetivos e apresenta-se o protótipo de suporte a esta tese.

O segundo capítulo aborda o estado arte no qual se procura fazer o enquadramento teórico sobre os principais temas abordados nesta dissertação.

O capítulo terceiro diz respeito ao sistema de testes. Neste, é detalhado o funcionamento do protótipo que se pretende desenvolver e é efetuada uma descrição pormenorizada sobre o material necessário.

O capítulo quarto diz respeito à especificação da aplicação e aborda os requisitos da aplicação. Para representar os requisitos da aplicação, em especial os requisitos funcionais, recorreu-se a diagramas de casos de uso da notação UML.

O capítulo quinto diz respeito ao desenvolvimento da aplicação no qual se pretende descrever todo o processo de desenvolvimento do sistema de testes que sustenta os conceitos apresentados na dissertação.

Finalmente, com o sexto capítulo, apresentam-se as contribuições principais do trabalho, os obstáculos e o trabalho futuro que se pretende desenvolver.

Capítulo 2

Estado da Arte

No capítulo dois é efetuado o enquadramento teórico sobre os principais temas abordados nesta dissertação.

2.1 Parapente

"You are not alone in your dream to fly. Through ancient tales and myths we know that humans from the beginning of time have longed for the freedom of the skies. Cavemen and women probably stood on a mountain ledge and envied the eagle soaring over the wooded landscape on outstretched wings"[44]

O Parapente é uma modalidade desportiva de voo livre que tem ganho adeptos em todo o Mundo. Apesar de ter algumas semelhanças com um paraquedas, na realidade é diferente em vários aspetos. O principal é a sua forma de voar, pois este pode descolar de uma encosta (montanha ou falésia), previamente "inflado" (aberto), ao contrário do paraquedas que se abre no ar após o salto do avião. É considerado como a forma mais económica e fácil forma de voar ao alcance do homem [5].

Trata-se de um desporto seguro desde que seja encarado de forma responsável. A frequência de um curso em instituição devidamente habilitada para esse efeito, assim como a realização de inspeções periódicas ao parapente são indispensáveis para garantir os níveis desejáveis de segurança. Não exige muito esforço físico, na medida que uma vez no ar, o utilizador "apenas" efetua a sua condução. O esforço maior é depreendido no momento do arranque em que é necessário efetuar uma ligeira corrida. O local de aterragem pode ser tão vasto quanto uns escassos 5 m2 ou pouco mais. Subindo em térmica podem-se atingir altitudes da ordem dos 3.000 metros, ou mais, descolando por exemplo de uma montanha de 200 metros de altura [43]. É interessante referir que se podem percorrer dezenas ou centenas de quilómetros a partir do local de descolagem.

O Parapente pode ser um misto de "relaxante" e "adrenalínico" sendo o piloto quem escolhe o que pretende. É relaxante na medida em que um piloto está sentado num arnês tipo "sofá" a apreciar a paisagem e adrenalínico se gostar de acrobacia (até *looping* é possível fazer em Parapente) [4].

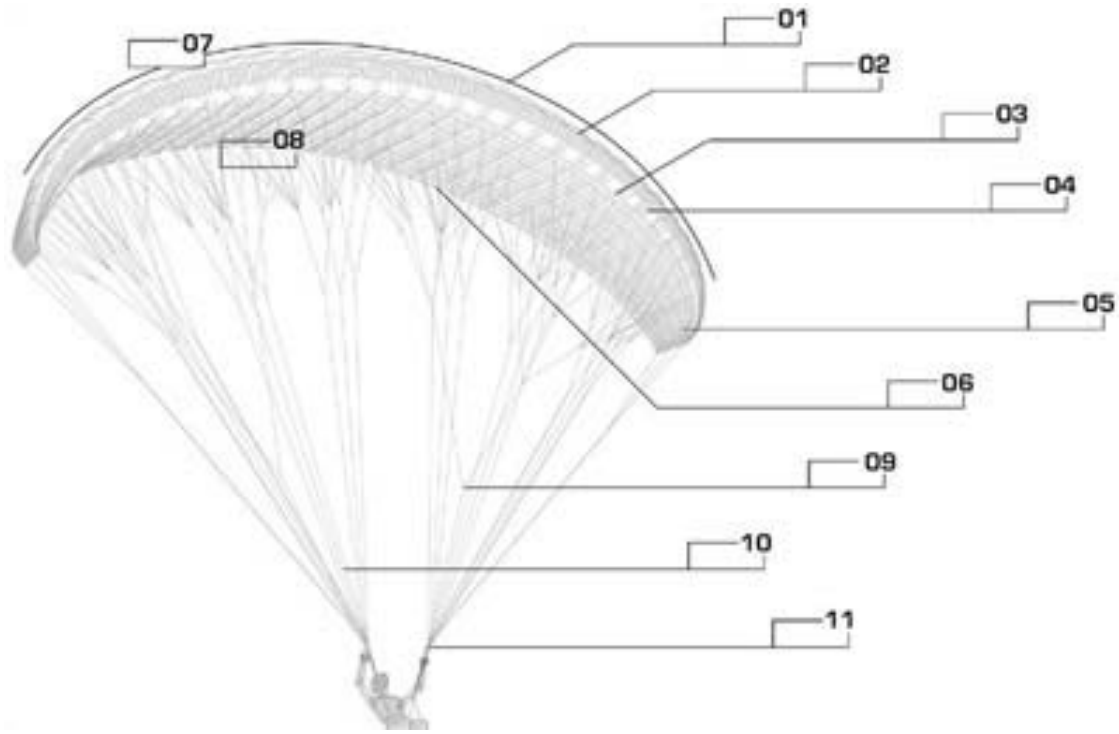


Figura 1 – Constituição de um parapente[30].

Na Figura 1 é possível observar a aparência geral de um parapente e alguns dos constituintes principais que o formam [44].

1. **Asa/Parapente/*Wing/canopy*** : Construída a partir de um tecido especial de *nylon* com um tratamento especial, que possui uma trama chamada *Ripstop* (que em caso de rasgo, impede que o mesmo aumente de tamanho)
2. ***Leading edge***: Bordo de ataque;
3. **Células/*Cells***: O parapente é dividido em secções individuais denominadas células;
4. ***Cell walls***: Aberturas por onde o ar entra nas células;
5. **Ponta da asa/*Wing tip***: É a ponta da asa também chamada de *estabilo;*
6. ***Trailing edge***: pé de galinha;
7. ***Top surface***: Extradorso;

8. ***Bottom surface***: Intradorso;
9. **Linhas de travão/*Brake line***. Ligada à parte traseira do parapente, permite que o parapente seja controlado;
10. **Linhas/*Lines***. Divididas em grupos (A, B, C, D), fabricadas usualmente em *Dynema* ou *Kevlar*;
11. **Tirantes/*Risers***. Conecta as linhas.

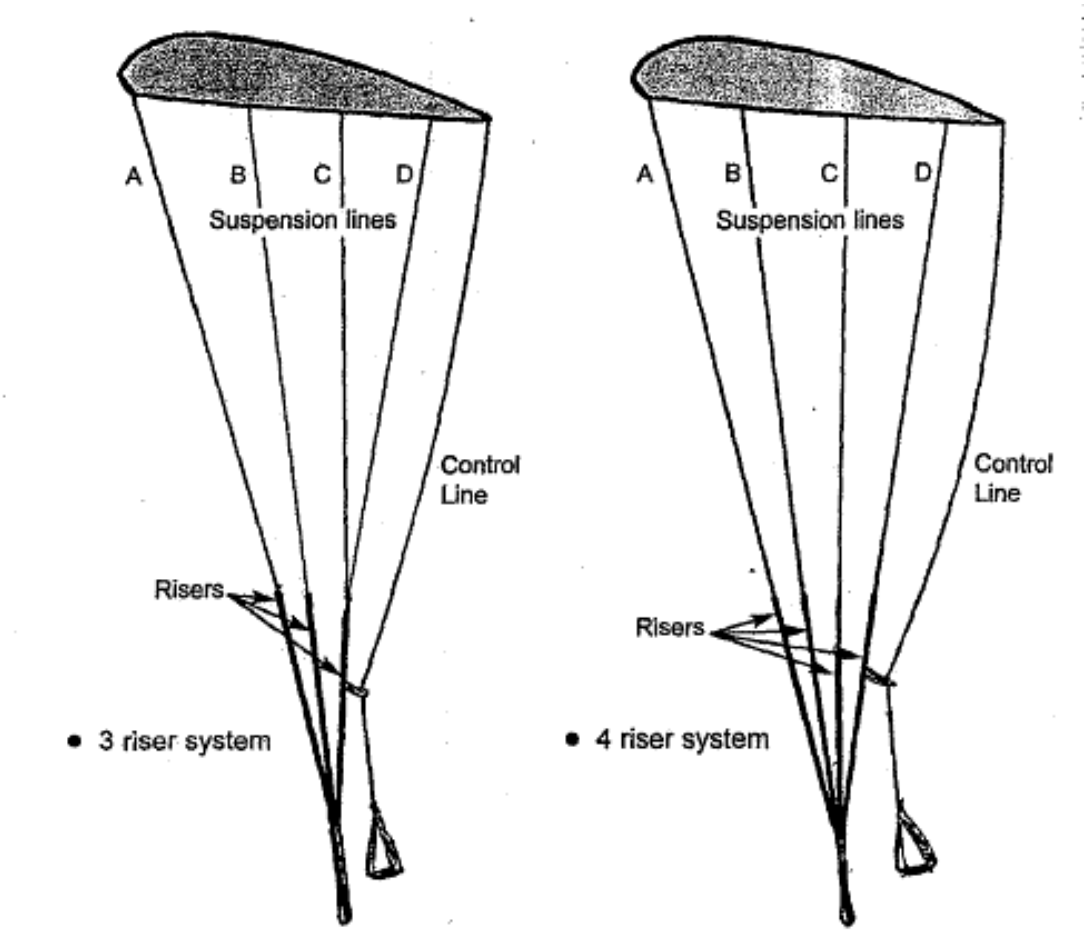


Figura 2 – Cone de Suspensão [44].

Observando em maior detalhe as linhas que formam o cone de suspensão de um parapente, verifica-se que a configuração mais usual utiliza três ou quatro tirantes como demonstrado na Figura 2.

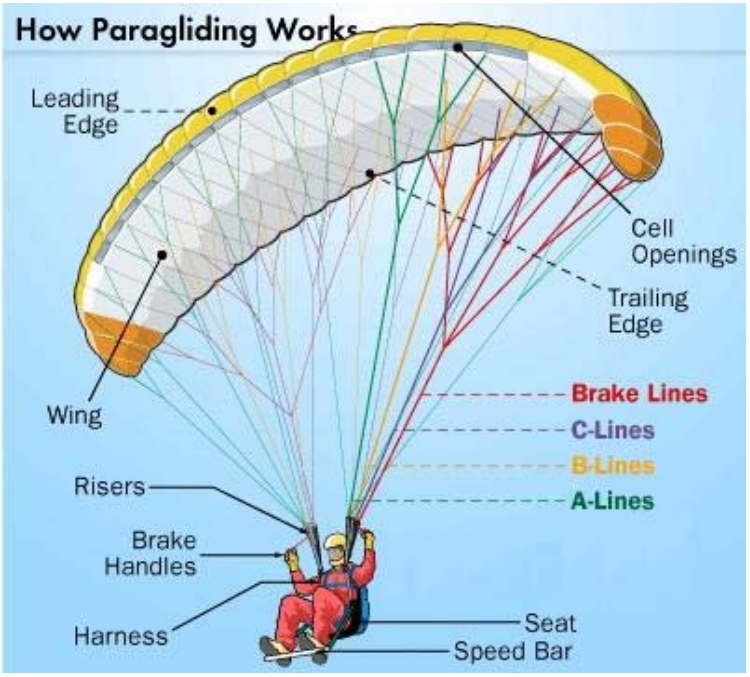


Figura 3 – Classificação das linhas[43].

Não obstante o número de tirantes do cone de suspensão, quase todos possuem quatro grupos de linhas que ligam o tirante à asa (*canopy).* As linhas são divididas em grupos mediante a posição que ocupam na asa como é possível comprovar na Figura 3. As que ocupam a posição da frente na asa e terminam nos tirantes da frente são denominadas de linhas do tipo A. As linhas seguintes são do tipo B, surgindo seguidamente as do tipo C e finalmente as do tipo D que ocupam a parte de traz [44].

Um parapente moderno tem entre 300 a 450 metros de linhas dispostas em cascatas que se encontram ligadas à asa [11]. Atualmente o número de linhas que constituem o cone de suspensão diminuiu, sendo que as linhas do tipo A foram reduzidas de cinco por lado para apenas duas ou três. Com este procedimento os fabricantes de parapentes obtêm menores custos, dado que não precisam de comprar tanta linha para o fabrico dos parapentes. No entanto isto obriga a que as linhas tenham que ser muito mais fortes do que eram antigamente.

A redução do número e da espessura das linhas tem também como objetivo aumentar a performance dos parapentes. Desde o início da produção de parapentes o consumo de linhas baixou 40% e a espessura reduziu de 4 milímetros para 2.1/1.6 mm, valor verificado na atualidade [11].

Figura 4 – Linhas parapente[11].

As linhas (Figura 4), embora sejam um produto de alta tecnologia, têm de ser cuidadas para se manterem em bom estado. A marca *Cousin* [20], muito conceituada no fabrico de linhas, aconselha as seguintes medidas no sentido de prolongar a vida útil do material:

- "Não deixe a sua asa perto de fontes de calor – no inverno pode ser o radiador, no verão o seu carro fechado."
- "Nunca guarde uma asa húmida ou num local húmido. Não seque a asa diretamente ao sol, coloque-a à sombra."
- "Não arraste a sua asa no chão."
- "Não deixe a sua asa aberta na descolagem durante muito tempo."
- "Evite torcer as suas linhas e não faça nós ou tranças para a arrumar."
- "Manobras (orelhas, espirais, fechos de asa e qualquer forma de acro) aceleram o processo de envelhecimento e enfraquecem as linhas. A realização frequente destas manobras requer que aceite as consequências – substituição mais frequente das linhas."
- "Depois de um grande choque (como um colapso muito grande) é necessária a verificação das linhas."
- **"Qualquer parapente usado deve ser submetido a uma verificação das linhas."**

- "Se as suas linhas têm proteção contra encolhimento pelo calor examine com todo o cuidado o limite do encolhimento pelo calor – a ponta leva a danos e fadiga."
- "Um parapente guardado que não é usado também envelhece."
- "Cuidado com linhas ultra finas ou sem camisa, especialmente em asas de competição. Estas requerem mais cuidado e danificam-se mais facilmente."

O método de fabrico das linhas segue as técnicas de construção usadas para o fabrico de cordas. Os primeiros parapentes que foram construídos utilizavam linhas feitas de poliéster entrelaçado. Alguns fatores como a sua elasticidade e pouca força contribuíram para que este fosse substituído como principal material de suporte de carga por outros tipos de materiais.

Os principais materiais que são usados atualmente são o *Dyneema* (polietileno de alta densidade), e *Technora*, *Kevlar* ou *Twaron*, todos nomes de marcas para poliamida ou *aramid[31]*. Nos princípios do parapente, toda a linha era feita de poliéster entrelaçado. No entanto, alguns fatores contribuíram para o fim do poliéster como o principal material de suporte de carga, entre os quais, a sua elasticidade e "pouca força" comparado com materiais mais recentes. É ainda usado no exterior (camisa) porque o exterior apenas suporta dez por cento da carga total da linha. Atualmente são utilizados dois tipos de material, estes são, o *Dyneema*, um polietileno de alta densidade, e *Technora*, *Kevlar* ou *Twaron*, todos nomes de marcas para poliamida aromática ou *aramid* [11].

Os princípios físicos que regem o voo do parapente são os mesmos que se aplicam a todas as outras aeronaves com asas. É a diferença de velocidade entre o ar que circula pelo extradorso e o intradorso que cria uma diferença de pressão gerando uma força ascendente (sustentação). Se esta força for superior ao peso da aeronave esta sobe[44].

Como se pode observar na Figura 5 o parapente usa o fluxo de ar nas superfícies do intradorso e extradorso para criar sustentação.

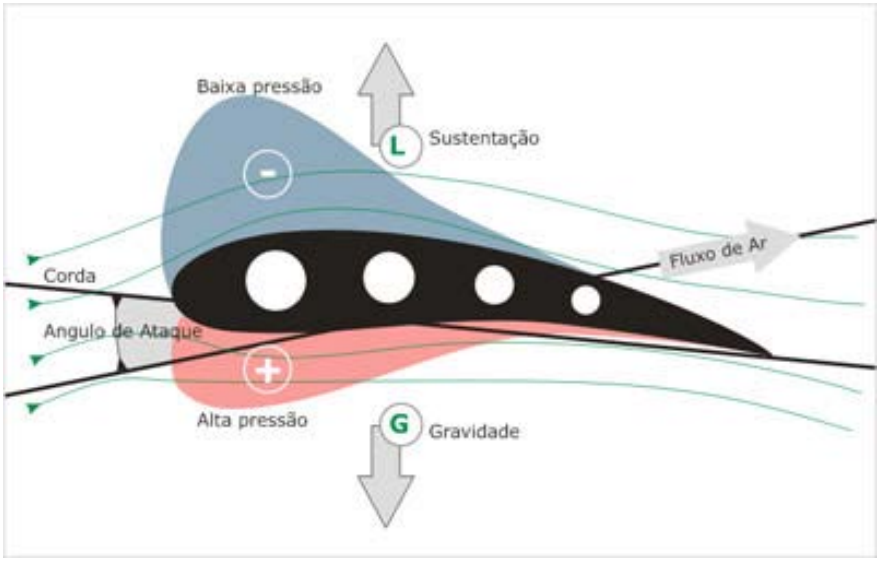


Figura 5 – Aerodinâmica da asa de um parapente [45]

Esta situação cria uma diferença de pressão entre as duas superfícies, a alta pressão do ar sob as asas tenta fluir em direção à baixa pressão do ar no topo e gera sustentação [45]. A título de curiosidade refere-se que este princípio é o mesmo utilizado nos carburadores dos automóveis.

2.2 Revisão Preventiva de Parapente

Para que o piloto de parapente possa garantir a sua segurança enquanto pratica este tipo de desporto, é essencial que efetue a revisão preventiva do seu equipamento. A revisão é importante não apenas no que diz respeito à segurança, como também contribui para um melhor comportamento e uma extensão da vida útil do equipamento.

Existem vários testes específicos adequados a cada um dos constituintes do parapente. É comum ser efetuada previamente uma inspeção visual tanto ao nível dos tecidos como dos fios que constituem o equipamento. Inicialmente deve ser cuidadosamente observada a parte superior e inferior da asa, procurando encontrar arranhões, buracos e outros danos [49].

Para avaliar de forma pormenorizada em que estado se encontra o tecido que forma a asa do parapente, efetuam-se dois tipos de testes. O primeiro está relacionado com a porosidade do tecido e é efetuado através de um equipamento como o da Figura 6, denominado “porosímetro”, que permite medir o grau de porosidade do tecido. O resultado do teste é dado pelo tempo (segundos) necessários para que 0,25 litros de ar passem através de uma área de tecido de 38.5cm2 sobre

uma pressão de 10mb. É então recalculada em unidades padrão e 20mb de pressão com a fórmula [31]:

permeability [l/m2/min] = 5400 / *measurement time* [s]

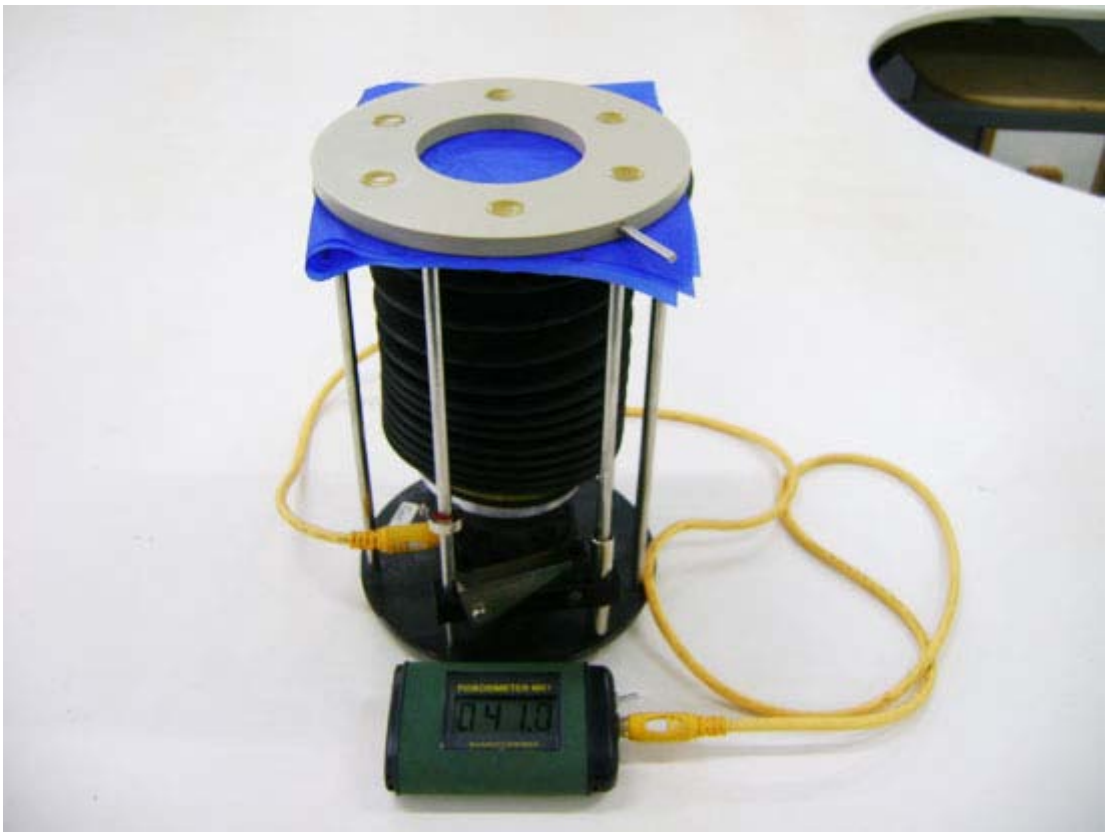


Figura 6 - Porosímetro

A medição é efetuada em diversas partes do parapente:

a. a superfície superior do Extradorso, em três células:

- Centro (metade da extensão da asa),

- Média da metade do Extradorso (¼ envergadura das asas),

- Última célula aberta

b. inferior do Extradorso, numa célula:

- Centro (metade da extensão da asa).

O dispositivo é montado a quinze por cento da altura da corda da asa (logo atrás da linha "A"). Como já foi referido, antes das medições, todo o tecido deve ser verificado, porque mesmo um pequeno dano pode alterar drasticamente os resultados. É aconselhável a observação do tecido contra a luz.

Os resultados são então recalculados para obter a média aritmética, que irá descrever a porosidade da asa. Se algum dos resultados for radicalmente diferente da

média, o local dessa medida deve ser cuidadosamente verificada outra vez e provavelmente terá que ser alterada.

O resultado do teste que é expresso em [l/m2/min] pode ser verificado na Tabela 1.

Tabela 1 – Dados de Referência Teste Porosidade [31]

0-20	"Como novo"
20-50	Excelente
50-100	Bom
100-150	Satisfatório
150-300	Tecido desgastado
Mais de 300	Tecido severamente desgastado. A segurança não pode ser garantida.

Quando o tecido é considerado como "desgastado", o protocolo deve declarar que a porosidade se aproxima do nível máximo aceitável. Se o resultado for superior a duzentos, a utilização do parapente pode ser estendida no máximo até doze meses. Quando o pano é considerado como "severamente desgastado", não é atribuído qualquer tempo de utilização já que este se encontra num estado em que não é possível garantir a segurança do utilizador.

Outro tipo de teste que deve ser efetuado à asa está relacionado com a resistência do tecido. Para efetuar este teste utiliza-se um equipamento de que dá pelo nome de "*Bettsometer*" e que pode ser observado na Figura 7 - *Bettsometer*[55].

Figura 7 - *Bettsometer*[55].

A medição é feita na superfície superior da asa em três células: centro (metade da envergadura da asas, ¼ da extensão da asa e da última célula aberta (mais próximo do estabilizador) [31].

O que é medido com este dispositivo é a força necessária para romper as fibras do tecido, por isso, na prática a sua resistência ao rasgo ao ser-lhe aplicada uma força mínima, predeterminada. Se o pano rasgar em qualquer um dos sítios da asa onde foi efetuado o teste, o parapente não deve ser utilizado, dado que a segurança está em causa. Na Figura 8 pode ser observado um exemplo de um teste à resistência de determinado tecido utilizando a ferramenta "*Bettsometer*".

Figura 8 – Utilização do *Bettsometer*[31].

Outro constituinte do parapente que deve ser tido em atenção é as linhas/cordões que formam o cone de suspensão do parapente. Inicialmente, à semelhança do que é feito com o teste realizado ao tecido da asa, deve ser efetuada uma inspeção visual à procura de danos. Para além desta, devem ser usadas as mãos para procurar qualquer alteração das linhas. Pelo que foi possível apurar, em Portugal, no que diz respeito aos cordões, estes são os únicos testes que são efetuados pelas empresa de reparação de parapente.

Figura 9 - Paramotor[5].

Este tipo de testes não são precisos e não garantem a segurança do utilizador do parapente. Como referido anteriormente, nos últimos vinte anos os pilotos têm optado por usar o parapente com motor (“paramotor”) dado que é mais versátil e apresenta uma maior facilidade de aprendizagem. É possível observar um “paramotor” na Figura 9. Apesar das vantagens referidas, o acréscimo do motor representa um aumento significativo ao peso geral da estrutura que forma o parapente. O peso acrescido vem reforçar ainda mais a importância que deve ser dada às revisões periódicas. Ao realizá-las, o objetivo é que o utilizador possa obter a garantia que todos os componentes do seu parapente estão em bom estado e que pode “voar” em segurança.

Os testes realizados pelas empresas especializadas na área do parapente são bastante rigorosos no que diz respeito ao tecido da asa, no entanto, relativamente aos fios que formam o cone de suspensão é efetuada apenas uma inspeção visual à procura de eventuais danos que sejam visíveis a “olho nu”. Um teste meramente visual não é rigoroso e não oferece as garantias mínimas de segurança. Sendo este um desporto em que se voa a altitudes elevadas, no que concerne à segurança deve ser adotado um critério muito rígido. Desta forma, é muito importante que as revisões efetuadas aos parapentes sejam melhoradas em alguns aspetos, nomeadamente no que diz respeito aos fios.

Seria uma mais-valia, para as empresas desta área, possuírem outro tipo de testes que possam diagnosticar de forma correta, o estado do do cone de suspensão do parapente. Apenas desta forma será possível garantir os níveis mínimos de segurança do piloto.

2.3 Sistemas de Aquisição de Dados

John Park e Steve Mackay definem aquisição de dados como: “o processo pelo qual os fenómenos físicos do mundo real são transformados em sinais elétricos que são medidos e convertidos em formato digital para análise, processamento e armazenamento por um computador.” [32]

Atualmente, o computador é a plataforma mais utilizada para efetuar sistemas de aquisição, processamento e tratamento de dados bem como o controlo de sistemas. Entre as principais razões para a sua utilização destacam-se o baixo custo, a facilidade de utilização, a flexibilidade e o seu desempenho.

Em muitas aplicações o que se pretende não é apenas a aquisição de dados, mas também efetuar ações de controlo sobre os sistemas em causa. O controlo

corresponde ao processo pelo qual os sinais digitais provenientes dos computadores são convertidos em sinais apropriados para atuar em diversos equipamentos de controlo: atuadores, relés, válvulas moduladores, entre outros. [32]

Os elementos básicos de um sistema de aquisição de dados, como mostrado na Figura 10 são:

- Sensores e transdutores
- Cabos de ligação
- Condicionadores de Sinal
- *Hardware* de aquisição de sinal
- *Software* de aquisição de sinal
- Computador (sistema operativo)

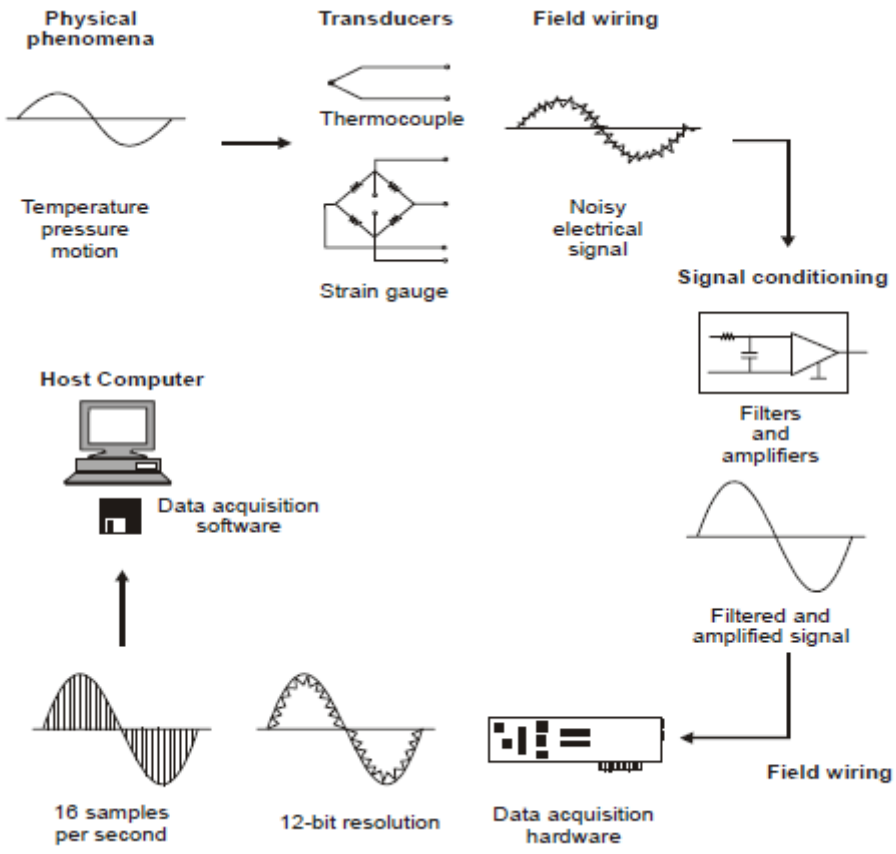


Figura 10 – Diagrama Sistema de Aquisição de Dados [32]

Cada um dos elementos que constituem um sistema de aquisição de dados apresenta elevada importância na medida em que afeta a precisão do sistema total de medição e a correta recolha dos dados do processo físico que se pretende monitorizar.

2.3.1 Sensores e transdutores

Os sensores e transdutores permitem que sinais captados no mundo real sejam convertidos em sinais elétricos (tensões ou correntes), apropriados para os sistemas de aquisição de dados.

Hoje em dia, existem sensores para medição de várias grandezas físicas. No caso de ser necessário efetuar a medição de temperaturas, existem os termopares, as Termo resistências, termístores e a junção de semicondutores, que convertem a temperatura do meio com o qual estão em contacto num sinal analógico proporcional; Para a medição de determinada carga exercida, existem as células de carga; Foram referidos apenas alguns dos sensores disponíveis, no entanto, como foi mencionado anteriormente, existem os mais variados tipos de sensores

Na Figura 11 é possível observar um sensor de deslocamento linear (Modelo *Unimeasure* P510 series) enquanto o equipamento da Figura 12 é um sensor de carga (Modelo LC101-2.5K).

Figura 11 – Sensor de deslocamento linear[33]

Figura 12 - Sensor de carga [48]

2.3.2 Cabos de ligação

Os cabos de ligação efetuam a ligação física entre os sensores e os condicionadores de sinal ou até aos equipamentos de aquisição de sinal no caso de não estarem incluídos no sistema total os condicionadores de sinal.

Quando o condicionador de sinal e/ou o sistema de aquisição é fisicamente afastado do computador, os cabos de ligação também fornecem a ligação física entre estes equipamentos e o computador. Nestas situações, estes cabos são vulgarmente designados como cabos de comunicação tal como sucede na comunicação RS-232 e RS-485. [33]

Dados que os cabos de comunicação representam na maior parte das ocasiões o maior elemento de todo o sistema, estes serão os mais suscetíveis aos efeitos externos como o ruído, que poderá afetar a precisão de medição.

2.3.3 Condicionadores de Sinal

Os sinais elétricos gerados pelos sensores e transdutores necessitam muitas vezes de ser convertidos numa forma apropriada para o equipamento de aquisição, particularmente para o conversor analógico-digital (A/D). Este, converte sinais elétricos em códigos digitais que poderão ser processados e armazenados pelos computadores.

Os sinais elétricos gerados pelos sensores e transdutores devem ser otimizados para a escala de entrada do conversor A/D, que converte sinais elétricos em códigos digitais que podem ser processados e armazenados pelos computadores.

As principais tarefas do condicionamento de sinal são: filtragem, amplificação, linearização, isolamento e alimentação.

• **Filtragem** – O propósito de um filtro é remover sinais indesejados do sinal que estamos a medir. Um filtro de ruídos é usado nos sinais DC, como temperatura, para atenuar sinais de alta-frequência que podem reduzir a precisão da medição. Sinais A/C, como vibração, geralmente requerem um tipo diferente de filtro conhecido como filtro anti-aliasing. O filtro anti-aliasing é um filtro passa-baixo que requer uma taxa de corte muito alta, e geralmente remove completamente todas as frequências do sinal que são maiores que a largura de banda de entrada do equipamento. Se esses sinais não forem removidos, eles irão aparecer erroneamente com os sinais da largura de banda de entrada do equipamento [32].

• **Amplificação** – O tipo mais comum de condicionamento é a amplificação. Sinais de baixa intensidade devem ser amplificados para aumentar a resolução e reduzir o ruído. Para uma maior precisão, o sinal deve ser amplificado de forma que a máxima tensão do sinal a ser condicionado coincida com a máxima tensão de entrada do conversor A/D [32];

• **Linearização** – Outra função comum do condicionamento de sinal é a linearização. Muitos transdutores, como os termopares, têm uma resposta não-linear às mudanças das ocorrências que estão a ser medidas [32].

• **Isolamento** – Outra característica comum no condicionamento de sinais é a isolação dos sinais dos sensores/transdutores em relação à entrada do conversor, visando um especto de segurança. O sistema a ser monitorizado pode conter sinais de

alta tensão que podem danificar o conversor. Uma razão adicional para a isolação é garantir que as leituras do equipamento de aquisição serão imunes a diferenças de potencial de terra ou tensões de modo comum. “Quando as entradas do sinal adquirido pelo dispositivo se referem a um potencial terra, podem ocorrer problemas se houver uma diferença de potencial em duas terras. Esta diferença pode levar ao que se chama curto de terra, causando imprecisão na representação do sinal adquirido; ou a diferença é tão alta que ela pode danificar o conjunto do sistema de medição. Usando módulos de condicionamento de sinal isolados elimina-se o curto de terra e assegura-se que os sinais são adquiridos com precisão” [32].

• **Multiplexagem** – Uma técnica comum para medir diversos sinais com um único equipamento de medição é a multiplexagem. O equipamento de condicionamento de sinal para sinais analógicos geralmente possui multiplexagem para uso com sinais de alteração lenta como temperatura. “O conversor A/D amostra um canal, troca para o próximo, amostra, troca para o próximo, amostra e assim sucessivamente. Por amostrar muitos canais ao mesmo tempo, a taxa de amostragem efetiva de cada canal é inversamente proporcional ao número de canais amostrados”. [32]

• **Excitação** – Alguns transdutores, requerem uma tensão externa ou sinais de corrente de excitação. Os módulos de condicionamento de sinal para esses transdutores geralmente geram esses sinais. Medições por RTD 8 fazem-se normalmente com uma fonte de corrente que converte a variação da resistência em relação a uma tensão mensurável [32].

Na Figura 13, apresenta-se um exemplo de condicionamento de sinal consistindo na amplificação do sinal elétrico original e sua filtragem para eliminar o ruído elétrico.

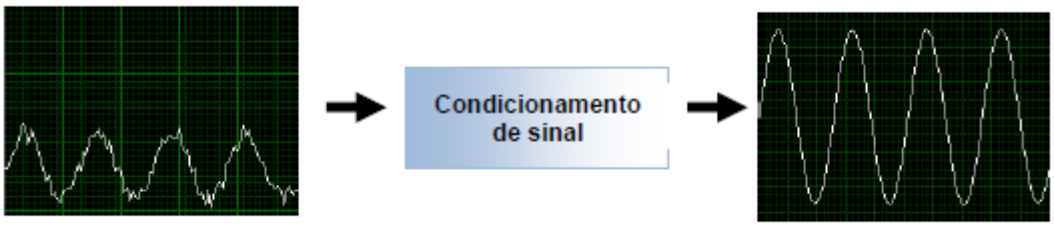


Figura 13 - Exemplo de condicionamento de sinal: amplificação e filtragem.

2.3.4 *Hardware* de aquisição de sinal

O *hardware* de medição é o responsável pelas entradas e saídas de sinais na cadeia de medida. Assim, ele pode executar qualquer uma das seguintes funções: [32]

- entrada, processamento e conversão para o formato digital, usando conversores digitais (AD), de sinais analógicos provenientes do meio de medição. Os dados após convertidos são transferidos para o computador para visualização, armazenamento ou análise;
- .entrada de sinais digitais que contêm informação acerca dum sistema ou processo;
- processamento, conversão para um formato analógico, utilizando conversores analógicos (DA) de sinais digitais do computador para controlo de processos;
- .saída de sinais de controlo digitais.

Existem diversos fabricantes que produzem este tipo de equipamentos, sendo que o utilizador deve analisar qual o mais indicado para o cenário onde o vai utilizar. Na Figura 14 e na Figura 15 é possível observar exemplos de equipamentos de aquisição de sinal.

O Arduíno é uma plataforma de *hardware* livre, projetada com um micro controlador Atmel AVR de placa única, com suporte de entrada/saída embutido e uma linguagem de programação padrão, na qual tem origem em *Wiring*, e é essencialmente C/C++ [7].

Figura 14 – Arduíno UNO [6]

Figura 15 – Velleman VM140 [52]

2.3.5 *Software* de Aquisição de Sinal

Para que um sistema de aquisição de dados funcione, este precisa da componente de *software,* já que é a partir desta que é possível controlar todo o processo de aquisição. Existem diversos tipos de *software* que permitem efetuar a aquisição de dados, desde os específicos para determinadas aplicações, a plataformas de desenvolvimento de aplicações de alto e baixo nível.

Como exemplo de um *software* de aquisição de sinal, refere-se o "Arduíno IDE" (Figura 16), que é uma aplicação multiplataforma escrita em Java na qual é derivada dos projetos *Processing* e *Wiring*. [7]

Figura 16 – Arduíno IDE

2.3.6 Computador

O computador incluído no sistema de aquisição de dados pode influenciar a velocidade à qual se pretendem adquirir os dados, o que poderá provocar uma diminuição da precisão, processamento e armazenamento dos dados.

Os componentes de um computador, assim como o seu sistema operativo, estão diretamente relacionados com os fatores referidos anteriormente. Assim deverá

observar-se com atenção o computador com que se pretendem fazer as medições, configurando-o sempre que possível para que não cause um impacto negativo nas medições.

2.4 Síntese

Para a que a realização desta dissertação fosse possível, foi necessário efetuar previamente uma pesquisa exaustiva sobre os temas a tratar. No capítulo dois é efetuada uma descrição sumária do estado da arte relativo aos seguintes temas:

- Parapente
- Revisão Preventiva de parapente
- Sistemas de aquisição de dados

Após análise deste capítulo é possível constatar que apenas as fábricas especializadas realizam testes específicos aos cordões que por estas são produzidos. Pelo que foi possível averiguar, não existe nenhum tipo de sistema no mercado que possa efetuar este tipo de testes.

Capítulo 3

Sistema de Testes

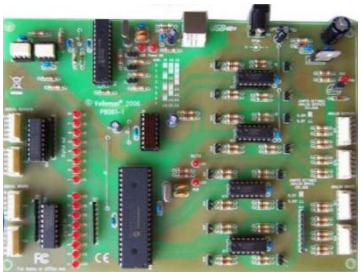
Neste capítulo pretende-se descrever o processo inerente à realização de testes ao cone de suspensão de um parapente. Inicialmente, em conjunto com a empresa, "Paraclinic" foi efetuada uma pesquisa no sentido de aferir de equipamento que permitisse a realização deste tipo de testes.

3.1 Equipamento necessário

- **Atuador eletromecânico:** Responsável por fazer a tração do cordão até rotura ou até atingir uma carga específica, a uma velocidade que não deverá exceder 30cm/min. A entidade colaboradora já possui um equipamento que cumpre com os requisitos.
- **Célula de carga**: Responsável por medir a carga a que o cordão está sujeita, em cada momento do alongamento do cordão.
- **Sensor de Deslocamento Linear:** Responsável por efetuar a medição do comprimento de um cordão ao longo de um teste.
- **Placas de interface sensores e PC**: A função destas placas está relacionada com a aquisição de dados e envio para um computador.
- **Condicionador de sinal:** As principais tarefas são: filtragem, amplificação, linearização, isolamento e alimentação.
- **Computador:** Deverá existir um computador com *software* específico que permita a recolha e tratamento de dados, assim como a emissão de relatórios de ensaio com grelhas de dados e gráficos.

Tendo em consta as especificidades apresentadas concluiu-se que o equipamento mais adequado seria o representado na Tabela 2.

Tabela 2 – Equipamento necessário para o Sistema de Testes

Equipamento	Função	Imagem
Omega – LC101-2.5K	Sensor de carga	 **Figura 17 – Omega LC101-2.5K [I.2]**
Unimeasure – P510-50	Sensor de deslocamento Linear	 **Figura 18 – *Unimeasure* P510-50 [I.1]**
Omega – IDRN-ST M2537	Condicionador de sinal	 **Figura 19 – Omega IDRN-ST M25377 [I.17]**
Velleman. VM140	Placa de aquisição de sinal	 **Figura 20 – *Velleman* VM140 [I.18]**

3.2 Testes

Os testes são efetuados a cordões e têm como objetivo aferir o estado do próprio cordão, assim como o estado geral de um cone de suspensão de parapente. Como referido anteriormente, um cone de suspensão é formado por diversos cordões interligados. O teste individual a cada um desses cordões seria um processo demasiadamente moroso. Embora fique sempre à consideração do utilizador do *sistema de testes* o número de cordões a testar, entende-se adequado obedecer ao seguinte processo:

- Inicialmente deve ser efetuada uma inspeção visual do estado dos cordões à procura daqueles que apresentem um maior nível de desgaste. N Figura 21 e Figura 22 é possível observar exemplos de danos que podem ocorrer em cordões de parapente.

Figura 21 – Possíveis danos num cordão (a)

Figura 22 - Possíveis danos num cordão (b)

- Idealmente deverão ser escolhidos para o grupo de amostragem quatro cordões de cada um dos tipos, nomeadamente tipo A, tipo B, tipo C e tipo D para a realização de testes. O objetivo é escolher os mais danificados de cada um destes tipos para a realização de testes.

Como explicado em capítulos anteriores, os cordões assumem papéis diferentes mediante a posição que ocupam na asa. Os cordões do tipo A ocupam a posição da frente na asa e estão sujeitos a um nível de carga mais elevado. Os do tipo C e D, dada a sua posição à retaguarda, sofrem cargas menores e portanto, não assumem um papel tão importante no que diz respeito à segurança.

A realização de testes a grupos de cordões ao invés de testes individuais traduz um maior grau de certeza nos resultados obtidos. Assim, pretende-se no final

de cada teste exibir um relatório relativo ao estado geral do cone de suspensão e não apenas de um cordão.

Após ouvir algumas pessoas com bastante experiência na prática desta modalidade (parapente), optou-se por incluir a possibilidade do utilizador poder atribuir "pesos" para cada tipo de cordão. Isto é, a cada tipo de cordão será dado um valor que traduz a sua importância no cálculo do resultado final. Esta parametrização é possível através da manipulação de valores contidos na tabela "ToleranciaCordao" (Figura 23), que faz parte da Base de Dados da aplicação desenvolvida.

ToleranciaCordao	
id_tc	int
tipoA	int
tipoB	int
tipoC	int
tipoD	int

Figura 23 – Tabela ToleranciaCordao

Os valores presentes na tabela poderão ser facilmente alterados por um utilizador através de um formulário disponibilizado na aplicação desenvolvida. Por defeito, a tabela apresenta os valores que é possível visualizar na Figura 23 e que expressam a maior importância dos cordões do tipo A e B e menor importância dos tipos C e D.

O desvio médio entre os dados obtidos durante um teste e os dados de fábrica do respetivo cordão é calculado através da seguinte fórmula:

A = {5,10,15,20,25,50,75,100,125,150,175,200,225}

i ∈ A

Max_i = máximo (A)

N = #A

$$\text{desvioMédioCordão} = \frac{\sum_{i}^{Max\_i} (dt_i - df_{i)}}{N}$$

A variável *i* representa os diferentes níveis de pressão a que determinado cordão está sujeito durante um teste. Expresso em daN.

A variável *dt* representa o alongamento do cordão em relação ao seu estado inicial quando a este é aplicada determinada carga (i). Diz respeito aos valores obtidos durante um teste e é expresso em percentagem (%).

A variável *df* representa o alongamento do cordão em relação ao seu estado inicial quando a este é aplicada determinada carga (i). Diz respeito aos valores de fábrica e é expresso em percentagem (%).

A variável N representa o número total de diferentes cargas que foram aplicadas ao cordão.

A fórmula utilizada para o cálculo do estado de um cone de suspensão de Parapente (estadoCSP), referente a quatro cordões do cone de suspensão do parapente que se pretende testar é:

$$\text{estadoCSP}(\%) = \frac{\sum (dmc_i . p_i)}{\sum p_i}$$

i= {A,B,C,D}

A variável *dmc* representa o valor obtido no cálculo do desvio médio de um cordão para determinado tipo de cordão (i).

A variável *df* representa o peso que foi atribuído para cada tipo de cordão.

Após cálculo do respetivo *estadoCSP* é possível classificar o estado do cone de suspensão que foi testado. Para isso é comparado o valor obtido com os valores presentes na tabela *ToleranciaMarca* (Figura 24).

id_tm	marca	MB	bom	suf	mau
1	cousin	0.1	0.2	0.3	0.5
2	Liross	0.1	0.2	0.35	0.4

Figura 24 – Tabela ToleranciaMarca

Como é possível observar na Figura 24, os valores poderão ser diferentes consoante a marca em questão. O utilizador do *sistema de testes* poderá alterar estes valores para outros que considere mais adequados tendo em vista uma melhor classificação do estado dos cordões. Os valores apresentados na Figura 24 são os que vêm por defeito na aplicação.

Para que fosse mais fácil de entender todo o processo que envolve a realização de um teste a um cordão, foi criado um fluxograma que pode ser observado na Figura 25.

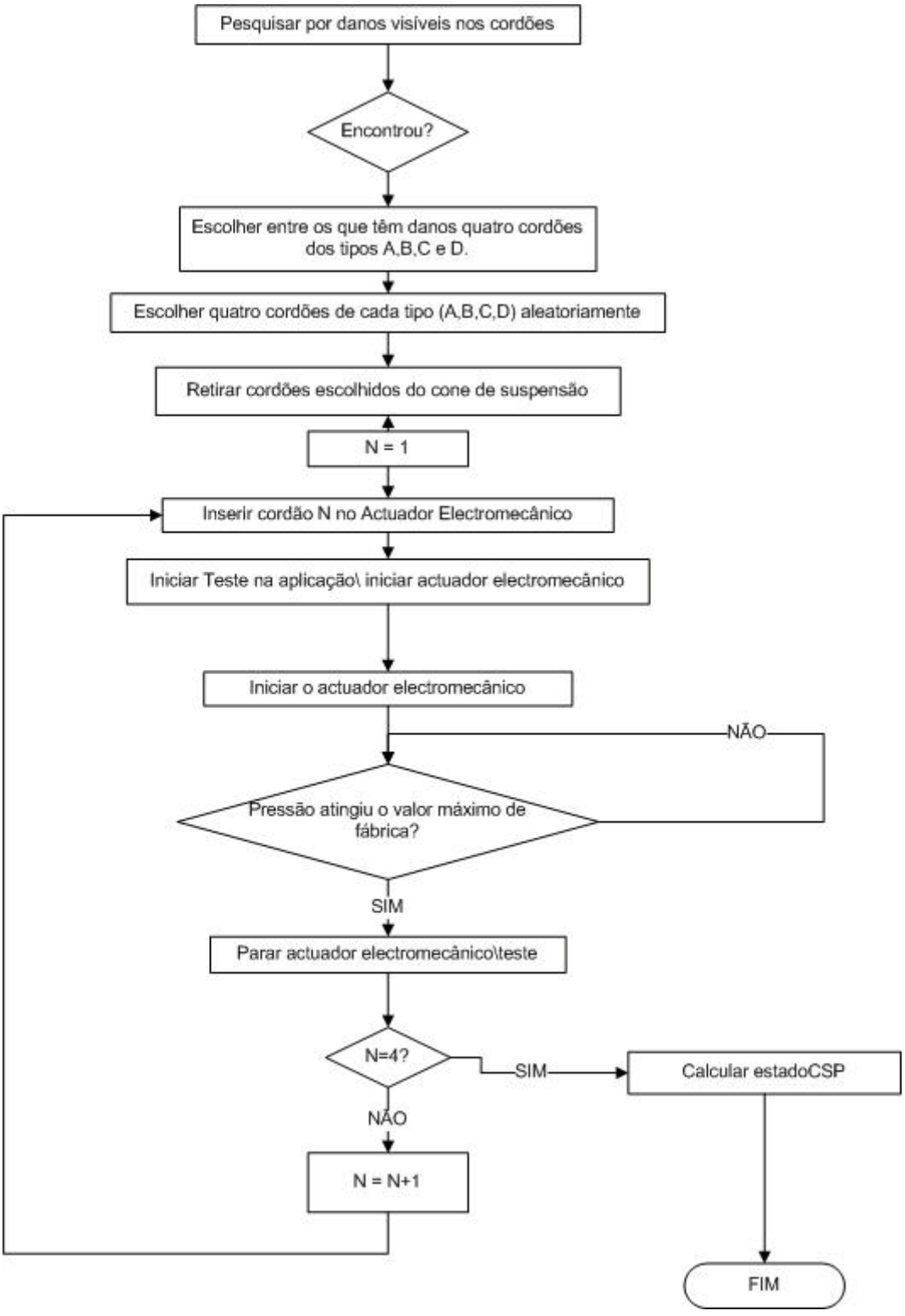


Figura 25 – Teste a um cordão (DFD)

A realização de um teste a um determinado cordão de uma asa executa-se usando um atuador eletromecânico, que faz a tração do cordão até determinado nível de carga, a uma velocidade que não deverá exceder 30cm/min. À medida que o cordão vai esticando, é medida a carga a que este está sujeito assim como o seu

alongamento em relação ao estado inicial. Estes valores são obtidos através dos sensores de carga e de deslocamento que se encontram instalados no sistema.

Os sensores deverão estar ligados a um condicionador de sinal que por sua vez irá transmitir os dados, depois de tratados, para o sistema de aquisição de dados. Este deverá transmitir todos esses dados para um computador adequado. Quando os dados chegam ao computador, o sistema deverá ser capaz de realizar diversos cálculos, assim como criar várias representações gráficas em tempo real.

Cabe ao utilizador interromper o teste, desligando o atuador eletromecânico e parando o teste na aplicação. Esta ação deve ocorrer antes que seja atingida a carga máxima do cordão em questão. Este valor encontra-se referenciado nas tabelas de fábrica de cada marca para cada referência de cordão em específico. (Anexo A)

Na Figura 26 é possível observar outro esquema relativo a um teste a um cordão, no qual se encontra representado o equipamento necessário.

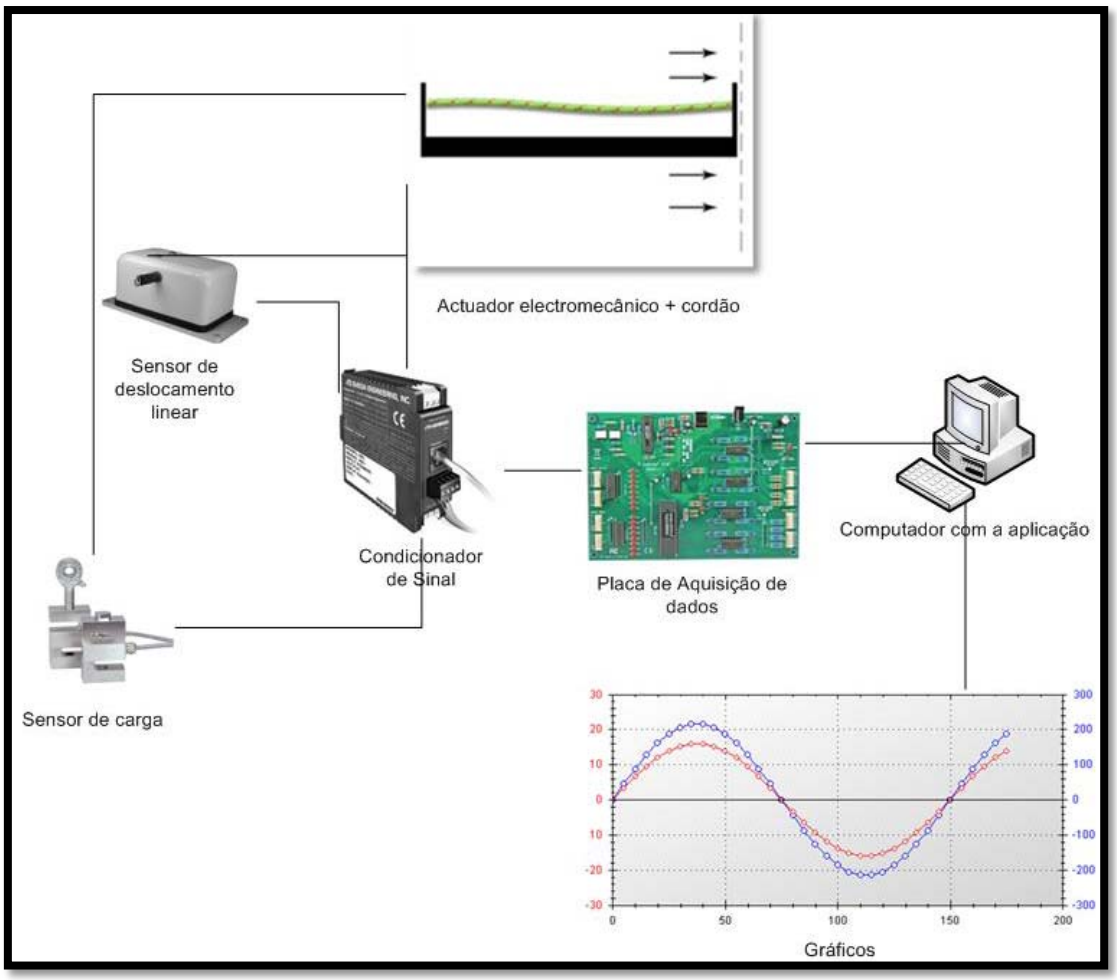


Figura 26 – Diagrama do Funcionamento de um teste

3.3 Síntese

Após a leitura do presente capítulo é possível entender o funcionamento geral de um teste executado a um cordão, assim como o material indispensável para a sua realização.

No próximo capítulo, irá ser utilizada essencialmente a linguagem de modelação UML, para descrever os requisitos da aplicação

Capítulo 4

Especificação da Aplicação

Neste capítulo serão especificados os requisitos de software do sistema. Será utilizada a linguagem de modelação de Casos de Uso (UML) com o objetivo de auxiliar a análise e gestão dos requisitos.

4.1 Requisitos da Aplicação

Com esta aplicação, pretende-se dotar as empresas especializadas na reparação de parapentes com um inovador sistema de testes. Usualmente, os requisitos de *software* são divididos em requisitos funcionais e não--funcionais. Assim, optou-se por efetuar uma descrição dos requisitos dividindo-os por esses grupos.

4.1.1 Requisitos Funcionais

- A aplicação deve possuir um interface gráfico que permita ao utilizador ter acesso a todas as funcionalidades do sistema.
- A aplicação deve disponibilizar um conjunto de funções para implementar o acesso a base de dados "*mysql*", permitindo selecionar, inserir, atualizar e eliminar registos de diversas entidades.
- Deve possuir também um conjunto de funções que permitam efetuar a ligação a um sistema de aquisição dados e receber valores provenientes de sensores.
- Durante a realização dos testes deverão ser exibidos gráficos relativos aos dados recebidos pelos sensores em função do tempo. No final deverá ser exibido um gráfico que permite comparar entre os dados obtidos no teste e os dados de fábrica dos cordões.
- Deve ser possível visualizar testes anteriores e respetivos valores e gráficos associados.
- Deve permitir o cálculo de valores como o desvio padrão e média.

- Deverá ser possível imprimir as tabelas com as listas dos elementos de várias tabelas, entre as principais, a tabela de cordões, testes, clientes e dados de fábrica.
- Deve possibilitar a emissão de um relatório final em que é exibido o estado do cone de suspensão do parapente.

4.1.2 Requisitos Não-Funcionais

- A introdução de dados no sistema deve estar protegida.
- A aplicação deve antes de submeter as instruções "*sql*", verificar se os dados estão em conformidade. Também na base de dados os dados deverão estar protegidos, sendo definidos em cada campo o tipo, o número de máximos de caracteres e a sua obrigatoriedade.
- A aplicação deve carregar os dados da base de dados referentes aos dados de fábrica, dados de testes e cordões para as listas correspondentes. Ao sair ou quando existir uma alteração esses dados devem ser inseridos na base de dados.
- O tempo de desenvolvimento da aplicação não deve exceder o tempo programado no cronograma de implementação.

4.1.3 Casos de Uso

Para documentar as diversas fases do processo de desenvolvimento da aplicação optou-se pela utilização uma abordagem de elaboração de sistemas de informação orientados por objetos. Assim, para representar os requisitos da aplicação, em especial os requisitos funcionais, recorreu-se a diagramas de casos de uso da notação UML.

Podem interagir com esta aplicação quatro tipos de atores, o utilizador, o SGBD (Sistema de Gestão de Base de dados), SCADA (*Supervisory Control and data aquisition*) *e* os vários sensores. Em seguida faz-se uma breve descrição de cada um destes atores:

- Utilizador – é o responsável dentro da empresa por efetuar a gestão do sistema. Terá acesso para efetuar todas as operações disponíveis no sistema. Terá a responsabilidade de definir os valores que achar adequados nas tabelas "TolerânciaMarca" e na tabela "TolerânciaCordão".
- SGBD – Entidade externa ao sistema onde ficarão armazenados todos os dados da aplicação.

- SCADA - Esta entidade tem como objetivo estabelecer a comunicação entre esta e o computador tendo em vista o envio dos dados recebidos pelos sensores.
- Sensor - os sensores permitem que sinais captados no mundo real sejam convertidos em sinais elétricos (tensões ou correntes), apropriados para os sistemas de aquisição de dados.

Tomando como referência cada um dos atores, nos pontos seguintes identificaram-se e descreveram-se os casos de uso em que participam.

4.1.3.1 Casos de uso: Utilizador/Base de Dados

O utilizador é um utilizador humano devidamente credenciado para efetuar a gestão e controlo da aplicação desenvolvida. Como tal, a sua interação com o sistema é efetuada numa lógica de utilizador - aplicação. Considerou-se a base de dados como sendo um elemento externo ao sistema e por essa razão foi adicionado como ator. Alguns atores consideram que a Base de Dados é um elemento do sistema, enquanto outros defendem que apresenta um papel dinâmico logo deve ser vista como ator.

O ator SGBD é um sistema de gestão de base de dados "*MySQL*" [40]. Neste, deverão estar registados todos os dados relativos a clientes, utilizadores, cordões, testes, dados de fábrica e tolerâncias. Também será usado para realizar determinadas pesquisas específicas.

Nos pontos seguintes estão definidos os casos de uso para o utilizador/Base de Dados:

- Gestão de clientes
- Gestão de Cordões
- Gestão de Dados de fábrica
- Gestão de Testes
- Gestão de Utilizadores
- Gestão de Tolerância Cordão
- Gestão de Tolerância Marca
- Iniciar Teste
- Parar teste
- Terminar sessão

Na Figura 27 estão representados os casos de uso nos quais o ator utilizador e base de dados podem participar.

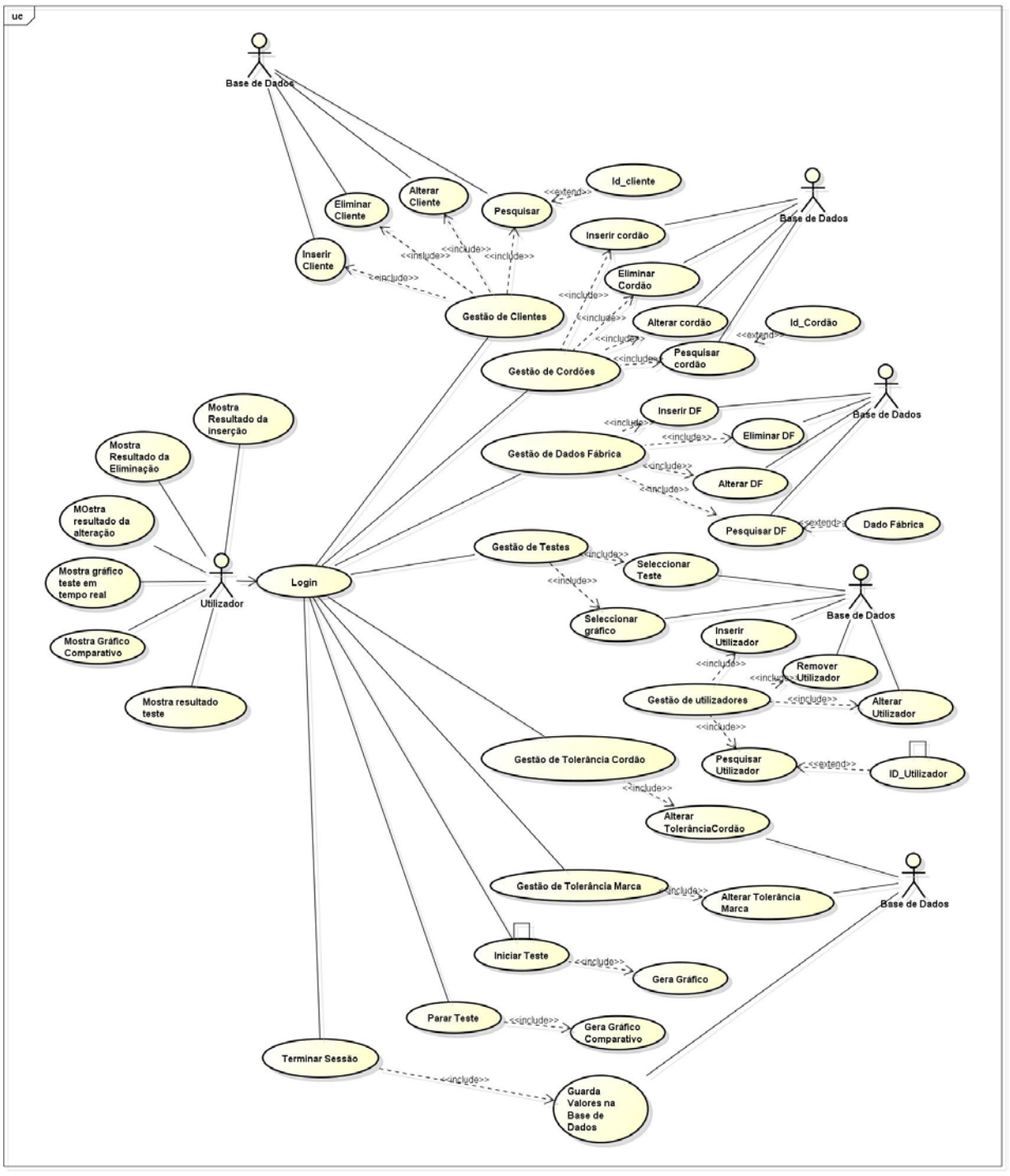


Figura 27 – Diagrama de casos de usos: Utilizador/ Base de Dados

Descrição dos casos de uso do utilizador/Base de Dados:

- Login
  - O/os utilizadores do sistema para acederem à aplicação deverão introduzir os dados que lhe foram fornecidos para se autenticar;
  - Deverão ser mostradas todas as opções da aplicação.
- Gestão de Clientes
  - O utilizador do sistema pode inserir, listar, alterar, pesquisar ou eliminar dados relativos a clientes. Também é possível imprimir a lista de todos os clientes.
- Gestão de Cordões
  - O utilizador do sistema pode inserir, listar, alterar, pesquisar ou eliminar dados relativos a cordões. Também é possível imprimir a lista de todos os cordões.
- Gestão de Dados de Fábrica
  - O utilizador do sistema pode inserir, listar, alterar, pesquisar ou eliminar dados relativos a dados de fábrica. Também é possível imprimir a lista de todos os dados de fábrica.
- Gestão de Testes
  - O utilizador pode visualizar todos os testes que se encontram registados na base de dados. Terá à sua disposição uma lista de gráficos para a apresentação dos dados de teste.
- Gestão de utilizadores
  - O utilizador do sistema pode inserir, listar, alterar, pesquisar ou eliminar dados relativos a utilizadores. Também é possível imprimir a lista de todos os utilizadores.
- Gestão de Tolerância Cordão
  - O utilizador do sistema pode alterar os valores predefinidos para a tolerância atribuída a cada tipo de cordão (Tipo A,B,C e D).
- Gestão de Tolerância Marca
  - O utilizador pode inserir, listar, alterar, pesquisar ou eliminar dados relativos a tolerância definidas para as marcas. Também é possível imprimir a lista de todas as tolerâncias atribuídas.
- Iniciar Teste
  - O utilizador pode a qualquer altura iniciar aceder ao painel de testes.
  - Antes de iniciar o teste terá de selecionar qual o tipo de cordão que vai testar. Este terá de previamente ser inserido na base de dados.
  - Assim que for selecionado o tipo de cordão, pode a qualquer altura dar início ao teste.
- Parar Teste
  - Cabe ao utilizador parar o teste assim que for atingido a pressão máxima tendo em conta o tipo de cordão que está a ser testado.
- Terminar Sessão
  - A sessão é terminada assim que o utilizador saia do programa

4.1.3.2 Casos de uso: Sistema de aquisição de dados/sensores

O sistema de aquisição de dados é um ator que, assim como o ator sensores, representa uma entidade externa ao sistema. O sistema de aquisição de dados tem o papel de transmitir e receber os dados provenientes dos sensores de e para o computador.

Nos pontos seguintes estão definidos os casos de uso para o utilizador/Base de Dados:

- Transmissão Dados PC
- Receção Dados PC
- Receção Dados Sensores
- Capta Dados

Na Figura 28 estão representados os casos de uso nos quais o ator utilizador e base de dados podem participar.

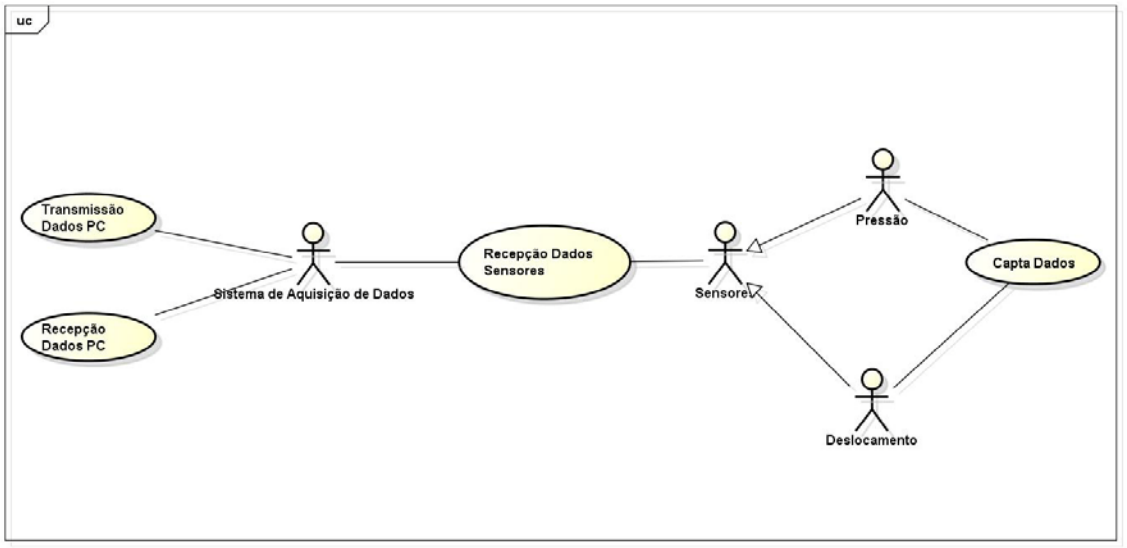


Figura 28 - Diagrama de casos de usos: Sistema de Aquisição de Dados/Sensores

Descrição dos casos de uso do Sistema de Aquisição de Dados/ Sensores:

- **Transmissão Dados PC**

o A placa de aquisição de dados deve enviar os dados recebidos pelos sensores que a este estiverem ligados através de uma porta serial para o computador. Mediante o protocolo que foi definido e que será explicado em detalhe no capítulo cinco, deverá enviar o valor devolvido pelo sensor de

pressão caso tenha recebido o carácter ‘A’ ou o valor registado pelo sensor de deslocamento caso tenha recebido o carácter ‘B’.

- **Receção Dados PC**
  - O sistema de aquisição de dados deverá receber através da porta serial o carácter que permitirá decidir qual dos valores dos sensores deverá transmitir como explicado anteriormente.

- **Receção Dados Sensores**
  - A placa de aquisição deverá receber os valores dos sensores de pressão e de deslocamento.

- Capta Dados
  - Devem ser captados os valores de pressão e de deslocamento relativamente ao teste efetuados ao cordão. Para isso são usados os sensores de pressão e deslocamento, no entanto poderão ser usados outros neste sistema.

4.1.4 Diagrama de Atividades

Para representar a dinâmica do sistema recorreu-se a diagramas de atividades da notação UML. Para cada entidade interveniente neste sistema computacional, isto é, para o gestor da aplicação e a aplicação cliente, apresenta-se inicialmente um diagrama geral que representa todo o fluxo de atividades envolvendo todos os casos de utilização da entidade.

Para os casos de uso que exijam mais detalhes devido à sua complexidade será apresentado um diagrama que descreve o fluxo de atividades.

4.1.5 Cliente e Base de Dados

No diagrama de atividades da Figura 30 estão representadas as ações de gestão e controlo da aplicação que serão executadas a pedido de um utilizador devidamente autorizado.

A aplicação foi desenvolvida utilizando uma arquitetura definida em três camadas, nomeadamente, UI (*User interface layer*), BLL (*Business Logic Layer*) e DAL (Data Acess *Layer*). De forma resumida, refere-se que a UI é constituída pelos formulários com os quais o utilizador do sistema interage. A camada BLL é responsável por armazenar a lógica da aplicação desenvolvida. Assim será nesta camada que serão efetuadas as validações respeitando as regras de negócio definidas. A camada DAL é responsável pelo acesso aos dados efetuando a comunicação entre a BLL e a UI [24].

O diagrama da Figura 30 pretende mostrar todas as ações disponíveis para os atores do sistema, utilizador e base de dados portanto, a sequência das ações pode ser qualquer após ser efetuada a validação do utilizador.

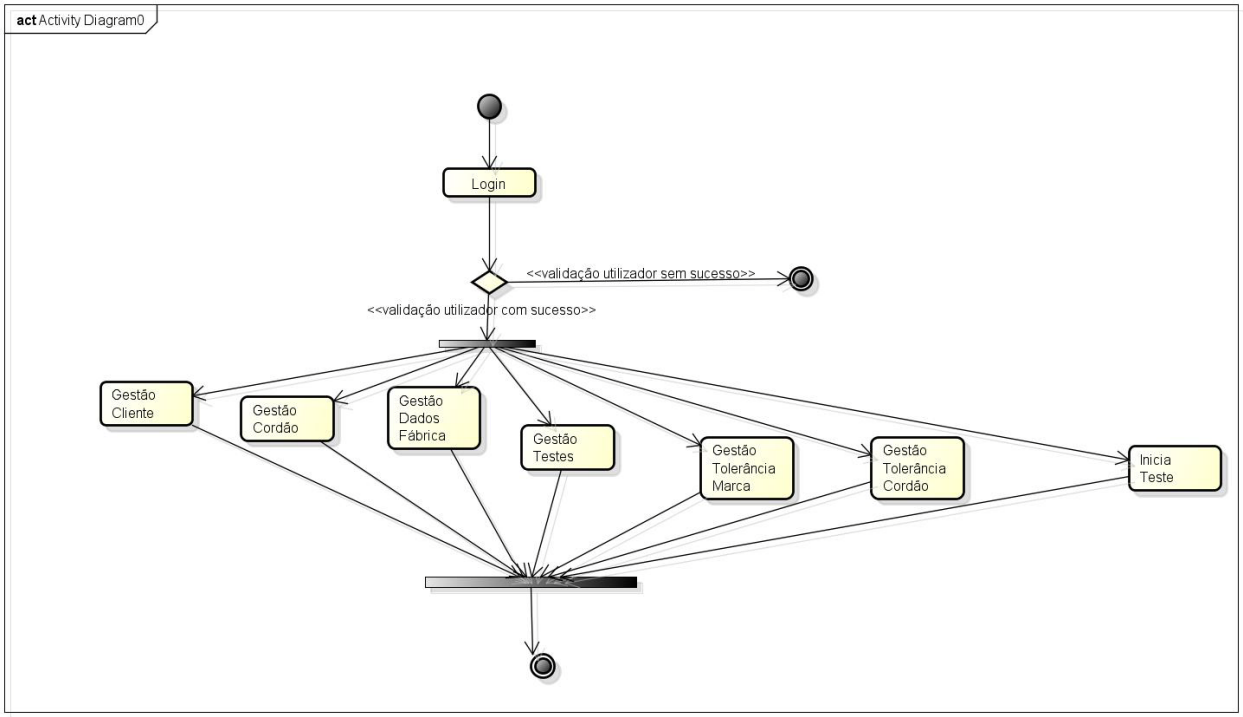


Figura 30 – Diagrama de actividades: casos de uso utilizador e base de dados

Na Figura 31 está representado um diagrama de atividades que decompõe em subactividades as atividades “GestãoCliente”, “Gestão Cordão”, “Gestão Dados Fábrica”, “Gestão Testes” ,”Gestão Tolerância Marca” e “Gestão Tolerância Cordão”.

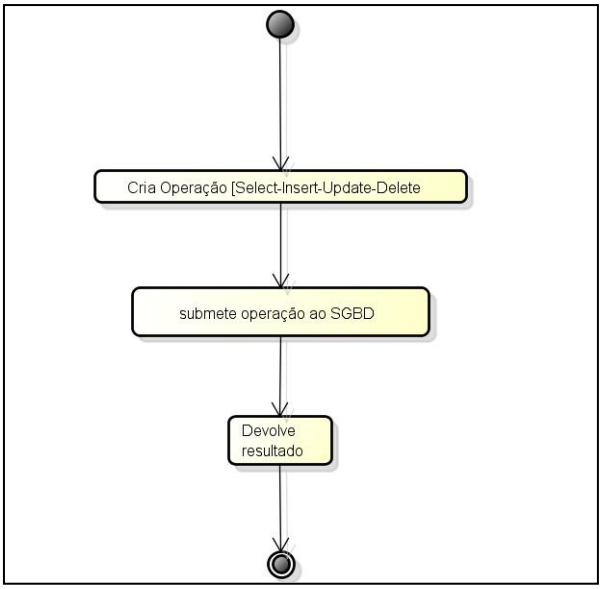


Figura 31 – Casos de uso “GestãoCliente”, “Gestão Cordão”, “Gestão Dados Fábrica”, “Gestão Testes” ,”Gestão Tolerância Marca” e “Gestão Tolerância Cordão”

Este diagrama descreve as atividades que são acionadas quando a aplicação cliente pretende selecionar, inserir, alterar ou eliminar registos de uma determinada tabela.

4.1.6 Diagramas de Sequência

Para representar o comportamento e a interação entre os objetos do sistema recorreu-se ao diagrama de sequência da notação UML.

4.1.6.1 Utilizador e Base de Dados

Na Figura 32, apresentada na página seguinte, encontra-se um diagrama de sequência que representa todos os casos de uso relativos aos atores utilizador.

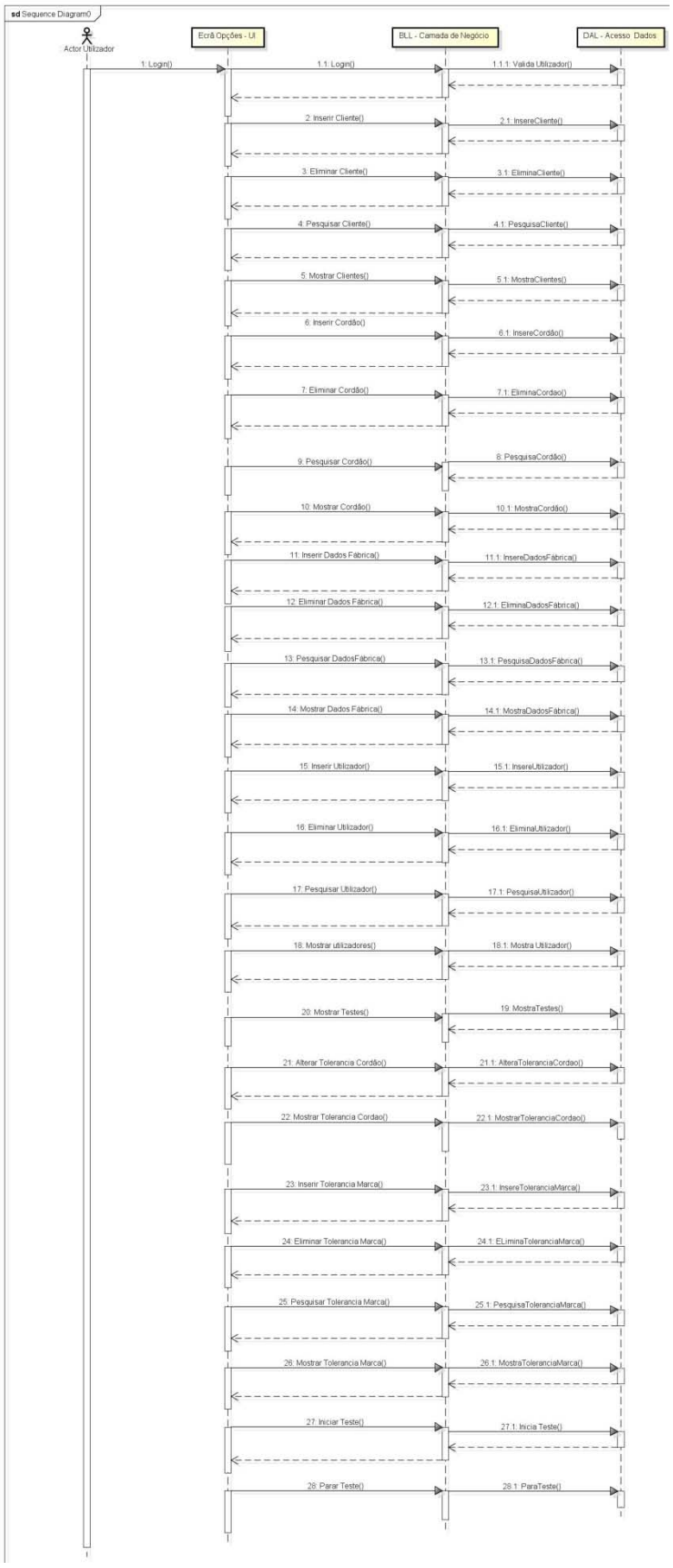


Figura 32 – Diagrama de sequência: casos de uso do utilizador e base de dados

4.1.7 Diagrama de classes

Os Diagramas de Classes mostram as diferentes classes que compõem um sistema e como elas se relacionam umas com as outras. Estes são apontados normalmente como “estáticos” uma vez que apresentam um conjunto de elementos estáticos tais como as classes, em conjunto com os seus métodos e atributos e os relacionamentos entre as mesmas [34].

Para o desenvolvimento correto da aplicação a modular, foram identificadas as seguintes classes:

Pessoa: representa a classe genérica definida para uma pessoa. No contexto da aplicação a pessoa pode ser um cliente ou um utilizador. A classe pessoa possui uma lista de elementos da classe cordao que expressa uma relação de composição.

Utilizadores: esta classe deriva da classe pessoa. É caracterizada pelos campos da classe mãe (Pessoa) e um identificador único (id_util), um *username*, uma *password* e um tipo.

Cliente - esta classe deriva da classe pessoa. É caracterizada pelos campos da classe mãe (Pessoa) e um identificador único (id_cliente).

ClienteDAL: representa a classe que efetua o acesso ao sistema de gestão de base de dados. É nesta classe que estão definidas todos os métodos que interagem com a base de dados da aplicação no que diz respeito à classe cliente.

ClienteBLL: representa a camada de negócios relativa à classe cliente. Nesta classe fazem-se todas as validações necessárias. É a partir desta classe, caso as validações respeitem as regras de negócio definidas para esta aplicação, que é chamada a classe “ClienteDAL” que por sua vez irá efetuar o acesso aos dados.

Cordao: representa um cordão que é inserido na aplicação. Possui um identificador único, id_cordao, e um conjunto de outros elementos que possibilitam caracterização de um cordão (Marca, referência, entre outros). Além disso possui uma lista de vários elementos da classe *dados de fábrica* e outra relativa à classe, *testes.*

cordaoDAL: representa a classe que efetua o acesso ao sistema de gestão de base de dados. É nesta classe que estão definidos todos os métodos que

interagem com a base de dados da aplicação no que diz respeito à classe cordao.

cordaoBLL: representa a camada de negócios relativa à classe cordao. Nesta classe efetuam-se todas as validações necessárias. É a partir desta classe, caso as validações respeitem as regras de negócio definidas para esta aplicação, que é chamada a classe *CordaoDAL* que por sua vez irá efetuar o acesso aos dados.

DadosFabrica: esta classe representa os dados de fábrica associados a um cordão. Possui um identificador único, id_df, assim como um conjunto de outros elementos como a constituição de um cordão, diâmetro e alongamento mediante determinadas pressões, entre outros.

DadosFabricaDAL: representa a classe que efetua o acesso ao sistema de gestão de base de dados. É nesta classe que estão definidos todos os métodos que interagem com a base de dados da aplicação no que diz respeito à classe *dadosfabrica*.

DadosFabricaBLL: representa a camada de negócios relativa à classe *dadosfabrica*. Nesta classe efetuam-se todas as validações necessárias. É a partir desta classe, caso as validações respeitem as regras de negócio definidas para esta aplicação, que é chamada a classe *DadosFabricaDAL* que por sua vez irá efetuar o acesso aos dados.

TesteCordao: esta classe tem como objetivo guardar os dados relativos a um teste realizado a um cordão.

TesteDAL: representa a classe que efetua o acesso ao sistema de gestão de base de dados. É nesta classe que estão definidos todos os métodos que interagem com a base de dados da aplicação no que diz respeito à classe *TesteCordao*.

TesteBLL: representa a camada de negócios relativa à classe *TesteCordao*. Nesta classe efetuam-se todas as validações necessárias. É a partir desta classe, caso as validações respeitem as regras de negócio definidas para esta aplicação, que é chamada a classe TesteDAL que por sua vez irá efetuar o acesso aos dados.

Comunicacao: esta classe representa a ligação entre o computador e o Arduíno. Possui um conjunto de métodos que permite estabelecer a ligação, enviar/receber dados entre o computador e o Arduíno e fechar a ligação:

printDGV: esta classe possui um conjunto de métodos e atributos que permitem que sejam criados modelos de impressão para os formulários.

LoginDAL: esta classe representa o acesso à base de dados tendo em vista a autenticação de utilizadores. Para aceder ao sistema de teste é necessário possuir um utilizador válido e respectiva palavra passe. Esta classe possui um método que é responsável por efetuar o acesso ao SGBD para verificar se os dados estão corretos.

LoginBLL: responsável por chamar a classe *LoginDAL*, tendo em vista a autenticação de um utilizador.

ToleranciaTipo: representa o peso atribuído a um tipo de cordão para a classificação final do estado de um cone de suspensão de parapente. Possui um identificador único, id_tp, e também os atributos tipoA, tipoB, tipoC e tipoD que permitem que o utilizador diferencie o peso de cada tipo de cordão.

ToleranciaTipoDAL: representa a classe que efetua o acesso ao sistema de gestão de base de dados. É nesta classe que estão definidos todos os métodos que interagem com a base de dados da aplicação no que diz respeito à classe *ToleranciaTipo*.

ToleranciaTipoBLL: representa a camada de negócios relativa à classe *ToleranciaTipo*. Nesta classe efetuam-se todas as validações necessárias. É a partir desta classe, caso as validações respeitem as regras de negócio definidas para esta aplicação, que é chamada a classe *ToleranciaTipoDAL* que por sua vez irá efetuar o acesso aos dados.

ToleranciaMarca: esta classe tem como objetivo permitir que existam diferentes classificações para as várias marcas. Isto é, no final de um teste a um cone de suspensão de um parapente é calculado um desvio final. Consoante o valor desse desvio será dada uma classificação ao cone de suspensão que pode ser: muito bom, bom, suficiente ou mau. Esta tabela permite que sejam configurados para cada marca valores para essas classificações.

ToleranciaMarcaDAL: representa a camada de negócios relativa à classe *ToleranciaMarca*. Nesta classe efetuam-se todas as validações necessárias. É a partir desta classe, caso as validações respeitem as regras de negócio definidas para esta aplicação, que é chamada a classe *ToleranciaMarcaDAL* que por sua vez irá efetuar o acesso aos dados.

ToleranciaBLL: representa a camada de negócios relativa à classe *ToleranciaMarca*. Nesta classe efetuam-se todas as validações necessárias. É a partir desta classe, caso as validações respeitem as regras de negócio definidas para esta aplicação, que é chamada a classe *ToleranciaMarcaDAL* que por sua vez irá efetuar o acesso aos dados.

Na Figura 33, apresentada na página seguinte, encontra-se o respetivo diagrama de classes relativo à aplicação desenvolvida.

4.1.7.1 Diagrama de classes

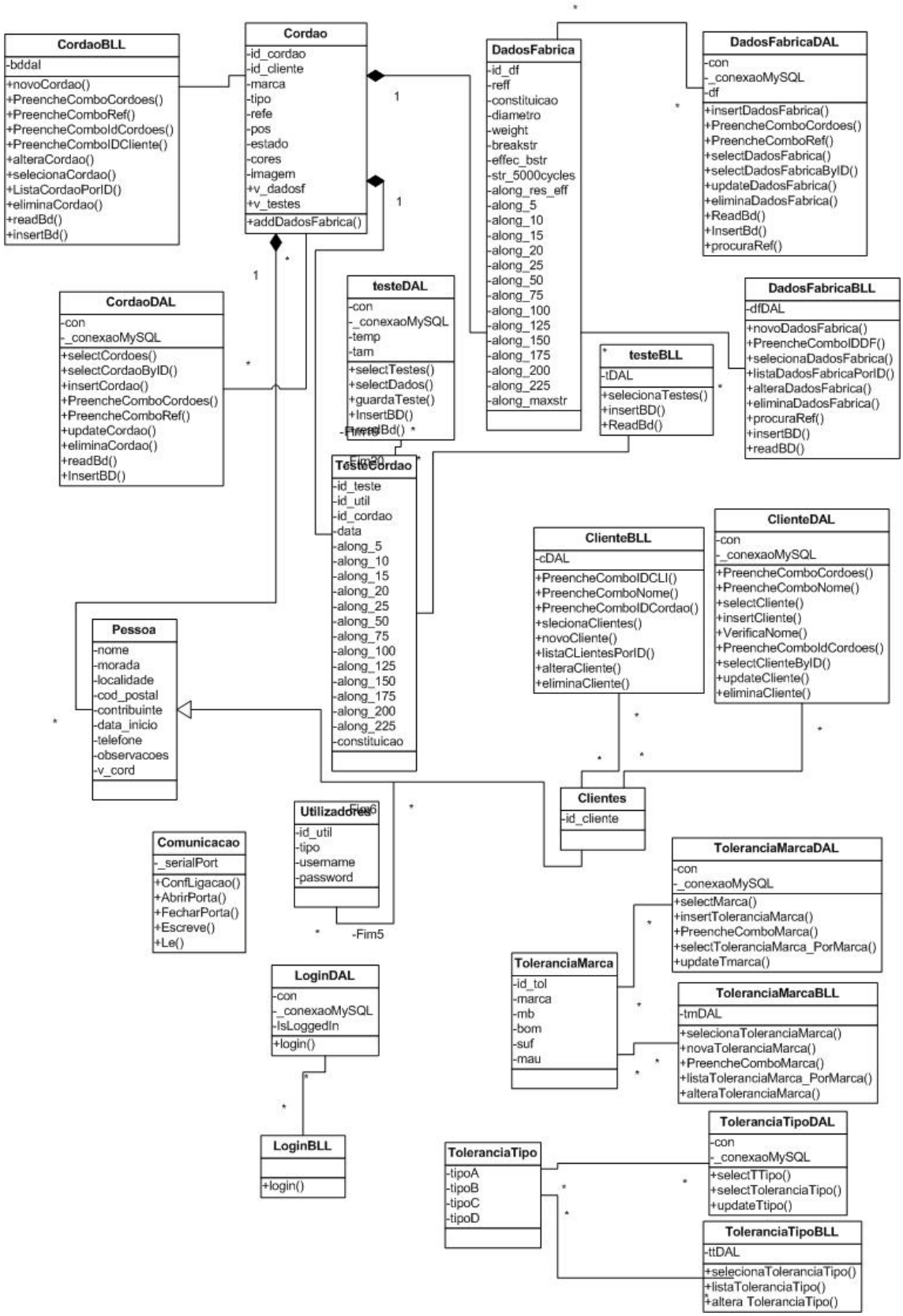


Figura 33 – Diagrama de Classes

4.2 Síntese

Durante este capítulo foram descritas as diversas fases do processo de desenvolvimento da aplicação através da representação de diagramas de caso de uso. Foi também efetuada uma descrição exaustiva das classes que compõem o sistema.

Seguidamente será efetuada a descrição do processo de desenvolvimento do sistema.

Capítulo 5

Desenvolvimento da Aplicação

No presente capítulo é efetuada uma descrição de todo o processo de desenvolvimento do sistema de testes que sustenta os conceitos apresentados na dissertação.

5.1 Introdução

Como foi referido anteriormente, o principal objetivo da aplicação desenvolvida consiste na criação de um sistema de testes que permita uma análise rigorosa do estado de um cone de suspensão de parapente. O processo de desenvolvimento desta aplicação foi orientado pelo modelo de desenvolvimento de *software* baseado em três camadas, UI, DAL e BLL.

O equipamento necessário para o desenvolvimento desta aplicação foi especificado no capítulo três, que diz respeito à descrição do Sistema de Testes. Até ao momento a empresa colaborante não adquiriu o material especificado e indispensável para a criação do sistema. Os custos associados à aquisição de todo o equipamento são relativamente elevados. Face a este obstáculo e para que fosse possível aproximar o desenvolvimento desta aplicação o mais próximo do real foi adquirido material alternativo. Desta forma efetuou-se a aquisição de um kit Arduíno e de sensores para colmatar esta dificuldade (Figura 34).

Foram utilizados sensores de temperatura (LW35 e LM33) que podem ser observados na Figura 35. A configuração de outro tipo de sensores será efetuada da mesma forma. A título de curiosidade refere-se que um dos sensores foi retirado de uma impressora antiga.

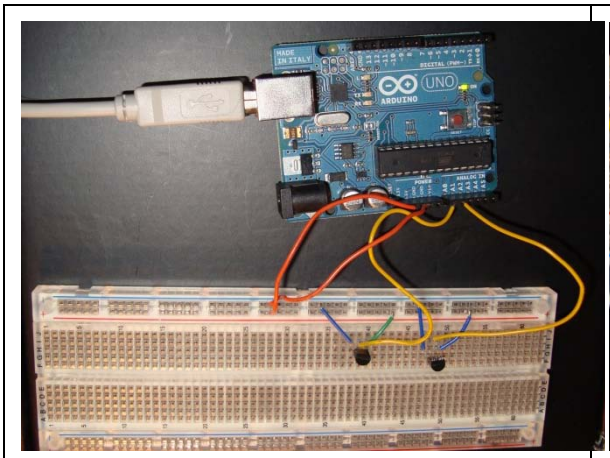


Figura 34 . Equipamento utilizado

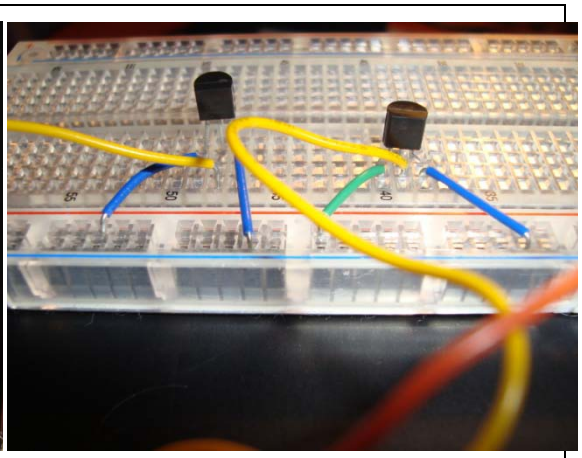


Figura 35 – Sensores utilizados

O desenvolvimento da aplicação baseou-se nas características do sistema de aquisição de dados Arduíno. No entanto, para que seja possível interagir com outros sistemas de aquisição de dados, a aplicação necessita apenas de ligeiras alterações. Dada a forma heterogénea de funcionamento entre os vários modelos de aquisição de dados, existem alguns detalhes que deverão ser tratados de forma distinta. Visto que inicialmente estava prevista a utilização de uma placa de aquisição de dados do tipo *Velleman VM140,* existe na aplicação uma classe que permite a comunicação entre esta placa e o computador.

Em termos de atividade de desenvolvimento da aplicação, esta será abordada subdividindo-a em quatro subsistemas principais:

- **Subsistema de gestão da aplicação** – a função principal deste subsistema consiste na criação de um interface gráfico que permita ao utilizador aceder às funcionalidades da aplicação de acordo com os requisitos identificados. Estas incluem as tarefas de gestão de:
  - Utilizadores;
  - Cordões;
  - dados de fábrica dos cordões;
  - testes
  - Tolerância Tipo Cordão
  - Tolerância Marca Cordão

- **Subsistema de acesso ao sistema de gestão de base de dados –** neste subsistema são implementadas todas as funções para acesso e manutenção do sistema de gestão de base de dados.
- **Subsistema de acesso e codificação da placa de aquisição de dados -** este subsistema tem como principal responsabilidade o estabelecimento da comunicação entre a placa de aquisição de dados e o computador. Incorpora também toda a codificação necessária para programar a placa de aquisição de dados (SCADA). O "Arduíno", que foi o SCADA escolhido para o desenvolvimento desta aplicação contém *software* específico onde será efetuada toda a codificação relativa à comunicação entre este e o sistema (computador). Possui também a responsabilidade de programar todos os sensores necessários.
- **Subsistema de criação de gráficos –** consiste na criação de gráficos que representam os valores do teste que se encontra em execução, assim como os valores relativos aos dados de fábrica do cordão que está a ser testado.

Qualquer um destes subsistemas foi desenvolvido com base nos requisitos levantados no capítulo quatro, quer nos funcionais, quer nos não funcionais, e nos casos de uso identificados para cada um dos utilizadores ou atores que podem interagir com o sistema.

5.1.1 Método de Desenvolvimento

Durante o desenvolvimento do capítulo anterior, relativo à especificação da aplicação, foi utilizada uma análise orientada a objetos. Neste capítulo é consagrado esse rumo tendo em conta o desenvolvimento da aplicação.

A programação orientada aos objetos assenta em três conceitos básicos fundamentais: encapsulamento de informação, composição/herança e polimorfismo [35].

A necessidade em proceder à separação de uma aplicação em camadas surgiu no início da década de noventa com os sistemas baseados na arquitetura cliente-servidor. Neste tipo de sistemas existem geralmente duas camadas, a interface gráfica e a camada de acesso aos dados (geralmente uma Base de Dados).

Na literatura é possível encontrar com frequência os termos *tiers* e *layers*, em inglês, que geralmente são traduzidos como camadas. Observando com mais atenção, embora a diferença seja subtil, compreende-se que *tiers* refere-se a uma separação

física dos componentes (diferentes *Assemblies*, *DLLs*, arquivos), enquanto *layers* significa uma separação lógica dos componentes com classes distintas e diferentes espaços de nomes [37].

Se o objetivo da aplicação a desenvolver é apenas a exibição e atualização de dados, este tipo de sistemas funcionariam corretamente. Tipicamente é necessário acrescentar regras de negócio, cálculos, validações, que vão originar alguns problemas. Estes problemas tendem a crescer proporcionalmente à medida que a aplicação se torna mais complexa. Perante este tipo de problemas, evoluiu-se para uma abordagem que utiliza várias camadas mediante as necessidades específicas de cada aplicação.

A parte mais difícil da utilização da arquitetura baseada em várias camadas reside em decidir quais as camadas que devem existir e qual a responsabilidade de cada uma delas. Geralmente fala-se no desenvolvimento de *software* respeitando a arquitetura de três camadas. Esta arquitetura contempla várias camadas separadas, nomeadamente a camada de apresentação, a camada de negócio e a camada de acesso aos dados.

- A camada de apresentação permite que o utilizador interaja com a aplicação. As responsabilidades primárias desta camada consistem na exibição de informação para o utilizador, assim como a interpretação dos comandos que este emite [24].
- A camada de negócio é responsável pela implementação das regras de negócio associadas ao problema e às entidades de negócio. É também responsabilidade desta camada implementar todas as validações de dados que necessitar.
- A camada de dados é responsável por toda a interação com os Sistemas de Gestão de Base de dados (SGBD) e outras fontes de dados.

Tabela 3 – Resumos Modelo 3 Camadas

Camada	**Responsabilidades**
Apresentação	Fornecimento de serviços, exibição de informações
Negócios	Lógica particular ao sistema
Dados	Comunicação com base de dados e outras fontes de dados

Existem algumas vantagens e desvantagens em desenvolver sistemas utilizando uma arquitetura em camadas que são descritas seguidamente: [37]

Vantagens:

- Reduzem complexidade: agrupam componentes e simplificam a comunicação entre eles;
- Reduzem dependência/acoplamento: a regra de comunicação evita dependências diretas entre componentes de Camadas diferentes;
- Favorecem a coesão: componentes de responsabilidades relacionadas são agrupados;
- Promovem reutilização: camadas podem ser reutilizadas em outros sistemas, podem ser substituídas;
- É um padrão arquitetural conhecido: facilita a comunicação e entendimento entre programadores.

Desvantagens:

- Limitadas pela tecnologia: algumas regras precisam ser infringidas dadas as limitações tecnológicas.
- Apenas complicam um sistema muito simples: não é qualquer sistema que exige o uso de Camadas;
- Possibilidade de “overdose”: muitos arquitetos de *software* acabam por criar Camadas a mais, tornando a aplicação extremamente complexa.

5.1.2 Plataforma de implementação

Para a implementação do sistema, foi escolhida a plataforma. *NET* da *Microsoft*. Para utilização em pleno da plataforma. *NET*, a *Microsoft* criou um ambiente de desenvolvimento integrado, o *Visual Studio .NET*, que contém um conjunto de ferramentas para a criação de todo o tipo de aplicações. A plataforma. *NET* representa uma mudança drástica nos métodos de desenvolvimento de *software* tradicionais da Microsoft. Toda a organização da estrutura da plataforma. *NET* é semelhante à plataforma Java, sendo as principais diferenças sentidas ao nível das camadas inferiores e superior [35]. A plataforma Java é multiplataforma ao nível de arquitetura de *hardware* e do sistema operativo, enquanto a plataforma. *NET* é dependente do sistema operativo Windows, embora existam atualmente alguns projetos para a implementar noutras plataformas tecnológicas. A nível de linguagens de programação a plataforma Java é dependente da linguagem Java, enquanto a plataforma. *NET* é multilinguagem, suportando entre muitas mais*, VB.NET*, *C#* e mesmo o *J#* com sintaxe Java, uma aplicação pode mesmo ser programada utilizando várias linguagens. A estrutura da plataforma. *NET* foi projetada para aceitar novas linguagens e novos compiladores. Portanto, a codificação do sistema foi realizada beneficiando das potencialidades da plataforma. *NET*, utilizando como linguagem de programação *C#.NET*.

5.2 Subsistema de gestão da aplicação

Este subsistema deverá permitir ao utilizador efetuar a gestão e controlo do sistema de testes. Para que isso fosse possível foi desenvolvida um aplicação gráfica que disponibiliza um conjunto de opções que permitirá ao utilizador usufruir de todas as funcionalidades da aplicação. Considerando que o sistema de testes foi desenvolvido respeitando a arquitetura em três camadas, este subsistema diz respeito à camada de apresentação.

O interface inicial da aplicação diz respeito à autenticação do utilizador. Por defeito foi criado na base de dados um utilizador inicial. Este, poderá após autenticar-se, efetuar a gestão de novos utilizadores Após devida autenticação surge o interface inicial do sistema (Figura 36) com as funcionalidades disponíveis.

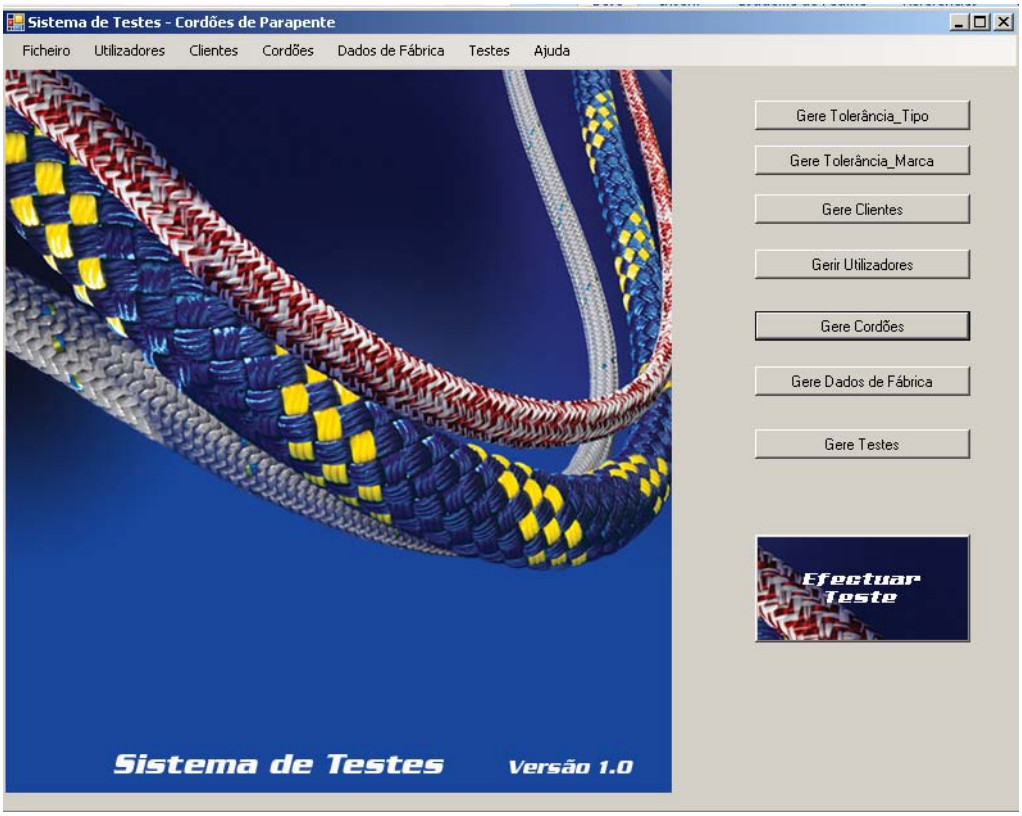


Figura 36 – Interface de Gestão

A partir deste interface é possível ao utilizador aceder a diversas funcionalidades, como a gestão de cordões, utilizadores, entre outras.

O objetivo principal da aplicação consiste na realização de testes a cordões. Para que isso aconteça é necessário que o utilizador execute previamente a inserção de alguns dados. Um teste é efetuado a um cordão que por sua vez possui dados de fábrica e pertence a determinado cliente. Assim, antes da realização de um teste é necessário que sejam criados clientes, cordões e dados de fábrica.

A criação de um cordão pressupõe a existência de um cliente associado. Da mesma forma, para que seja possível comparar o resultado de um teste realizado a um cordão é necessário que previamente tenham sido inseridos na base de dados os dados de fábrica relativos ao modelo do cordão testado.

A gestão das diversas entidades (cliente, cordões, entre outras) realiza-se de forma idêntica. Desta forma optou-se por efetuar apenas a descrição sobre a forma como é possível efetuar a gestão de uma entidade, dado que para as outras será um processo idêntico.

Na Figura 37 é possível observar o interface que surge quando o utilizador seleciona a opção “Gere Cordões.” O utilizador pode efetuar diversas tarefas como inserir, alterar, eliminar e mostrar os cordões existentes.

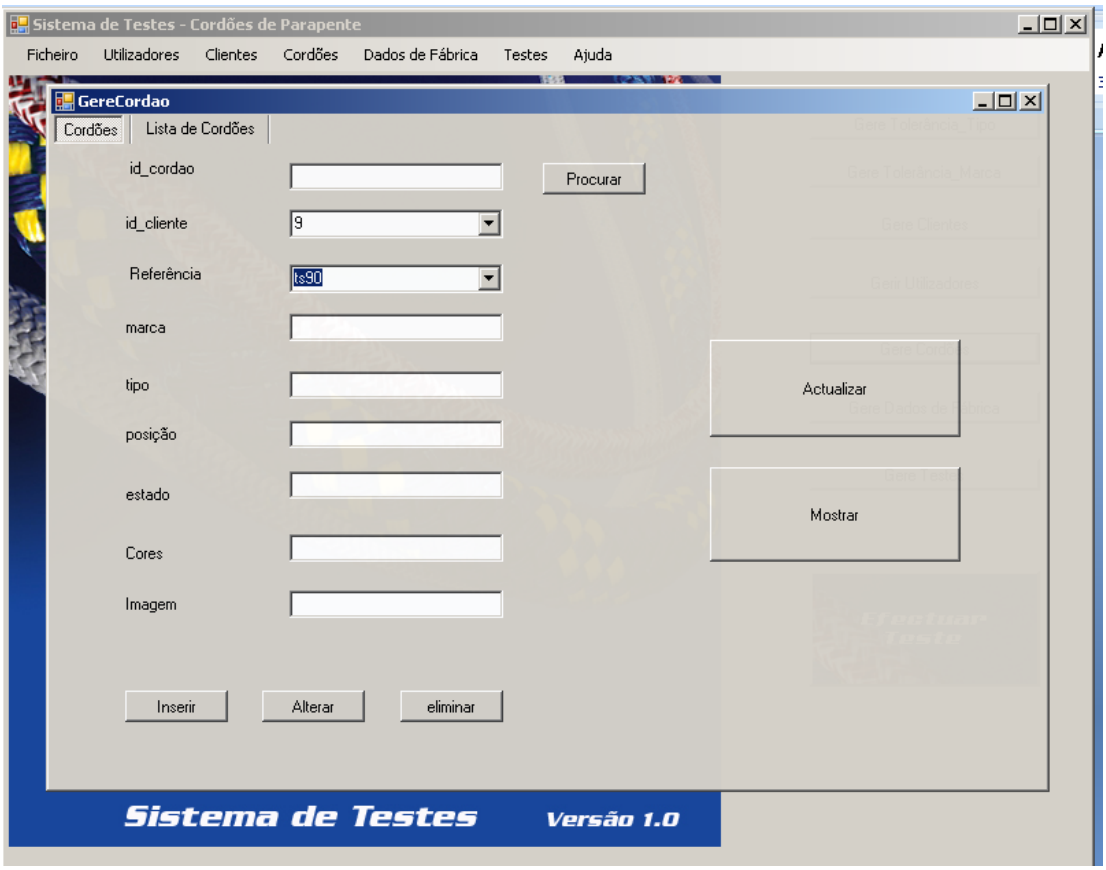


Figura 37 – Interface de Gestão de cordões

Como referido anteriormente, a inserção de um cordão pressupõe a existência de clientes e dados de fábrica. Para inserir um cliente o utilizador deverá selecionar obrigatoriamente qual o cliente que lhe pertence e qual o modelo do cordão que pretende inserir.

Para proceder à alteração\eliminação de determinado cordão deverá ser efetuada uma pesquisa pelo cordão e depois apenas carregar no botão com a função pretendida.

Para visualizar uma listagem de todos os cordões que se encontram inseridos na base de dados o utilizador deverá selecionar a opção “Mostrar” e seguidamente escolher o separador “Lista de Cordões”. Após esta seleção será mostrada uma tabela com a respetiva lista de cordões (Figura 38).

Figura 38 – Lista de cordões

Uma vez neste interface, o utilizador poderá efetuar a impressão da listagem de todos os cordões. Na Figura 39 é possível observar que o utilizador poderá atribuir um título para a impressão assim como escolher quais os campos que deverão ser exibidos na impressão

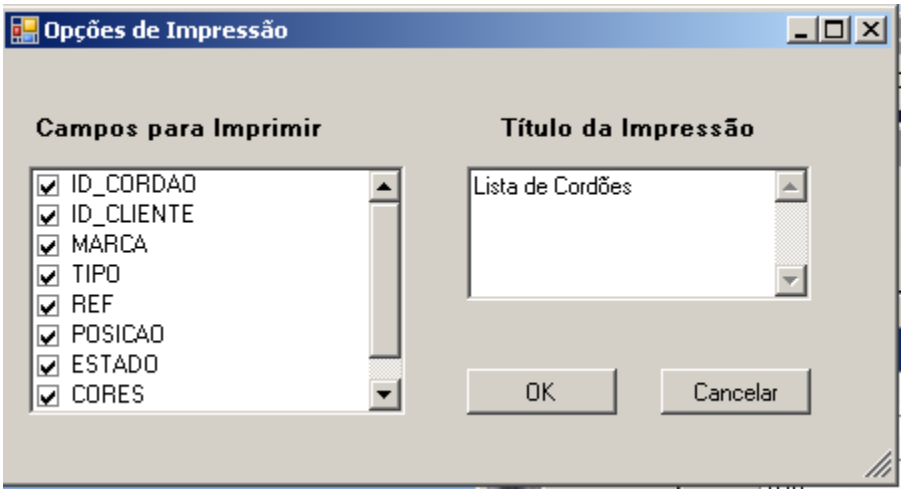


Figura 39 - Opções de Impressão

Pressupondo que a base de dados da aplicação já contém clientes, cordões e dados de fábrica, o utilizador pode optar por realizar um teste. Após selecionar a opção “Efetuar Teste” será possível visualizar o interface da Figura 40. Antes de iniciar o teste o utilizador deve escolher qual o cliente e cordão respetivo. Após esta ação poderá dar início ao teste.

No decorrer do teste serão gerados gráficos para auxiliar o utilizador a obter uma melhor perceção dos dados obtidos.

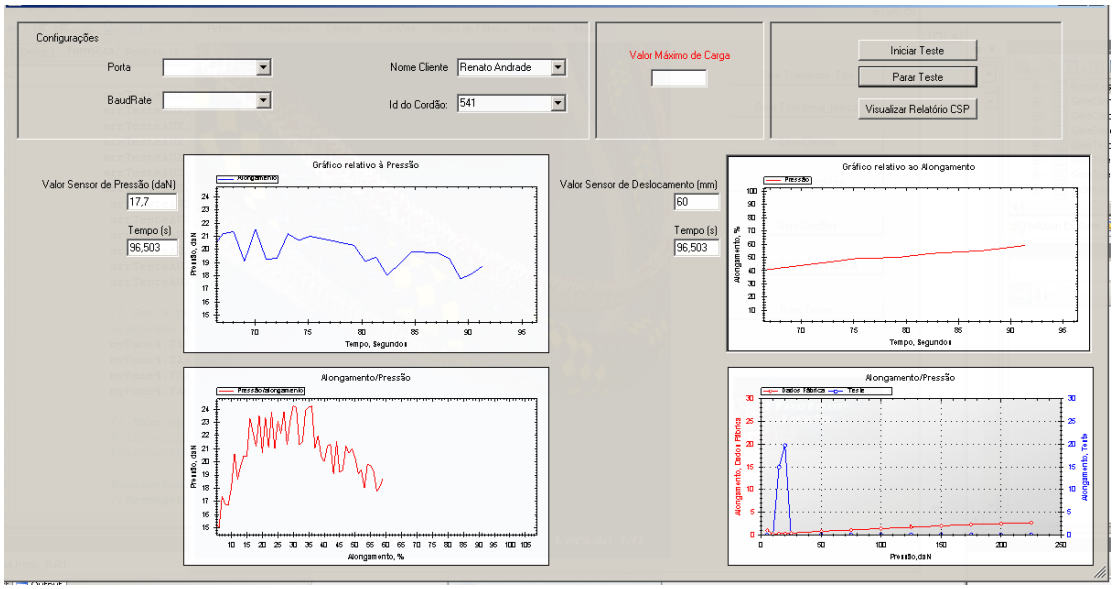


Figura 40 – Interface de Testes

Cabe ao utilizador terminar o teste assim que entender ou quando for atingido o limite máximo de carga. Dado que este valor é diferente para os diversos modelos de cordões existentes no mercado, será exibido no interface, o respetivo valor como pode ser observado na Figura 40.

Uma análise aos gráficos gerados permite aferir do estado do cordão que foi testado. Caso se pretenda analisar o estado de um cone de suspensão de parapente, deverão ser efetuados quatro testes, de acordo com o especificado no capítulo três, “Sistema de Testes”, e no final deve ser selecionada a opção “Visualizar relatório CSP”.

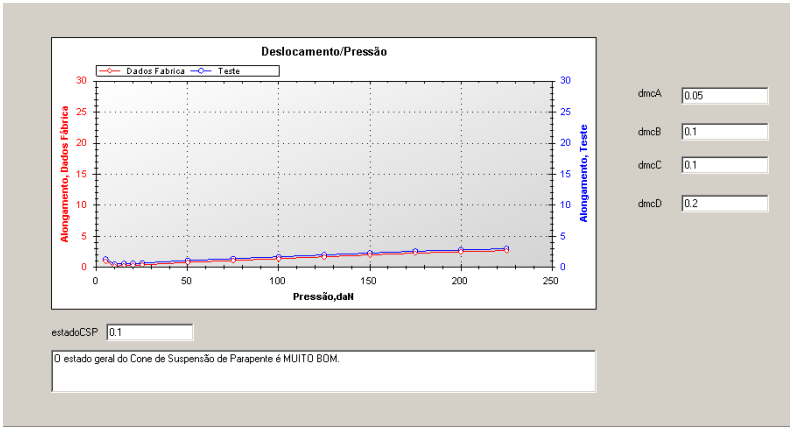


Figura 41 – Gráfico relativo ao estado geral de um cone de suspensão

A Figura 41 representa o interface relativo ao estado geral de um cone de suspensão de parapente. Mediante os testes obtidos a aplicação vai dar uma classificação relativa ao estado do cone de suspensão.

5.3 Subsistema de acesso ao sistema de gestão de base de dados

Este subsistema é responsável pelas operações de acesso e manutenção da Base de dados do sistema de testes. Representa a camada de acesso aos dados (*Data Acess Layer* - DAL).

Sempre que no decorrer da aplicação seja invocada uma operação ao sistema de gestão base de dados através da camada de aplicação *(User Interface* - UI), esta camada será a responsável por satisfazer o pedido. No entanto, como já referido anteriormente, esta chamada não será efetuada diretamente, terá de passar obrigatoriamente por uma camada intermediária denominada de "BLL" (*Business Logical Layer*). Esta camada irá invocar a camada "DAL" caso todas as validações efetuadas respeitem as regras de negócio previamente estabelecidas.

O sistema de gestão de base de dados escolhidos foi o "mySql". Esta escolha deve-se ao facto de se tratar de um *software* livre com base na *GPL*, pouco exigente a nível de recursos de *hardware* que apresenta um excelente desempenho e estabilidade. [40].

5.3.1 Análise de Dados

5.3.1.1 Entidades

Entidade Cordao:

A entidade cordão representa os dados relativos a cada cordão que se pretende submeter a um teste. Antes de ser iniciado um teste a um cordão é necessário que este tenha sido previamente inserido na base de dados do sistema de testes. Fazem parte desta entidade os atributos:

- **Id_ cordão** (numérico): número que identifica cada cordão de forma inequívoca. Pode conter no máximo oito algarismos e é incrementado automaticamente.
- **Marca** (caracteres): representa a marca do cordão possuindo no máximo vinte caracteres.
- **Ref** (caracteres): todos os cordões inseridos na base de dados deverão possuir uma referência. Todas as marcas associam referências aos cordões de forma a ser possível distingui-los. Por exemplo, a marca "*Liros*" apresenta entre outras, as seguintes referências de cordões: "Liros LTC45", "DFL115". Através da referência é possível identificar

dados específicos associados ao cordão como o diâmetro, peso, entre outros. Possui no máximo vinte caracteres.

- **Posição** (caracteres): classificação do cordão mediante a posição que ocupa no cone de suspensão de parapente. Os valores possíveis são: TipoA, TipoB,TipoC e TipoD.
- **Estado** (caracteres): classificação do cordão quanto ao estado do cordão. O estado pode assumir os seguintes valores, “novo”,”usado”.
- **Cores** (caracteres): campo destinado a armazenar as cores de determinado cordão possuindo no máximo vinte caracteres.
- **Imagem** (caracteres): campo que representa o caminho para a imagem relativa a um cordão. Pode ter no máximo cinquenta caracteres.

A entidade cordão tem relacionamento com:

- A entidade Clientes com o nome de relação “tem” com cardinalidade N:1 com obrigatoriedade na entidade cordão

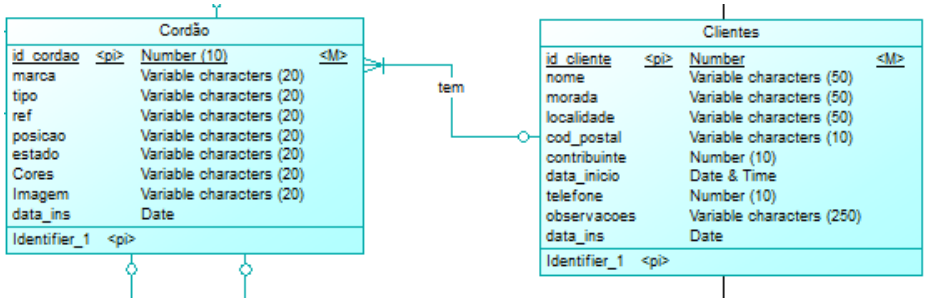


Figura 42 – Relacionamento entre as entidades cordao e clientes

- A entidade Teste com o nome de relação “efetua” com cardinalidade 1:N com obrigatoriedade na entidade teste.

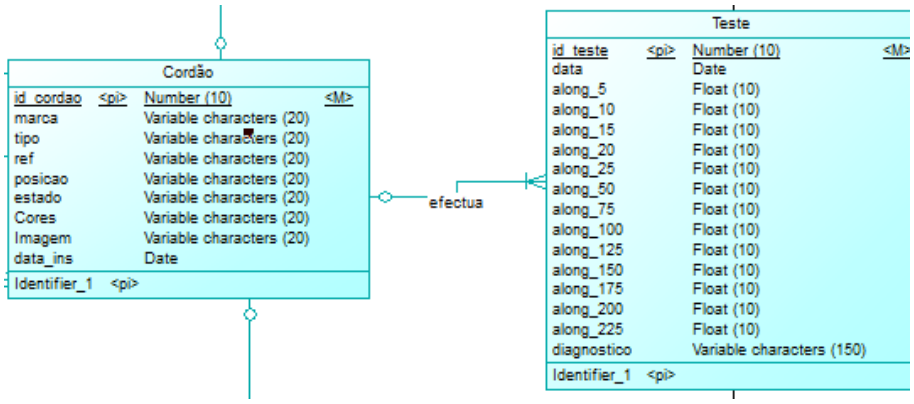


Figura 43 – Relacionamento entre as entidades cordao e teste

- A entidade DadosCordoes com o nome de relação “origina” com cardinalidade 1:N.

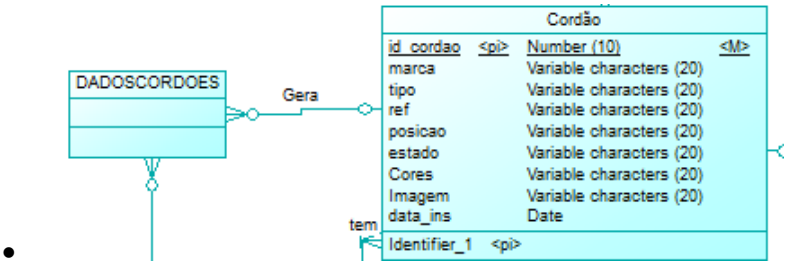


Figura 44 – Relacionamento entre as entidades cordao e dadoscordoes

- A entidade ResultadoTeste com o nome de relação “possui” com cardinalidade 1:N

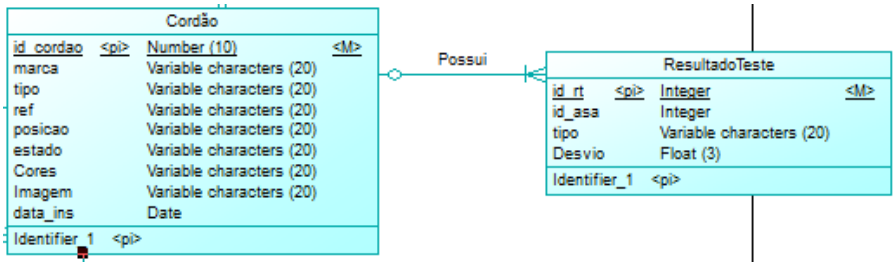


Figura 45 – Relacionamento entre as entidades cordao e resultadoteste

Entidade Clientes

A entidade clientes refere-se aos dados relativos a cada cliente que entrega um cone de suspensão para ser alvo de reparação. Para cada cliente deverá ser inserido um registo na base de dados. Fazem parte desta entidade os atributos:

Id_cliente (numérico): número que identifica cada cliente de forma inequívoca. Pode conter no máximo oito algarismos e é incrementado automaticamente.

Nome (caracteres): representa o nome completo do cliente. Exemplo: Renato Filipe Martinho Andrade

Morada (caracteres):campo destinado a armazenar a morada do cliente.

Localidade (caracteres): campo destinado a armazenar a localidade do cliente.

cod_postal (caracteres): representa o código postal completo do cliente.

Contribuinte (caracteres): representa o número de contribuinte do cliente.

Data_inicio (caracteres): este campo destina-se a armazenar a data em que o cliente foi registado na base de dados do sistema de testes.

Telefone (caracteres): representa o número de telefone do cliente.

Observações (caracteres): este campo destina-se a armazenar informações que o utilizador entenda importantes relativamente ao cliente que está a inserir na base de dados do sistema de testes.

A entidade cliente tem relacionamento com:

- A entidade cordão com o nome de relação “tem” e com cardinalidade 1:N com obrigatoriedade na entidade cordão(Figura 42).

Entidade Teste

A entidade Teste tem como objetivo armazenar os valores obtidos durante um teste a determinado cordão. Estes dados são armazenados na base de dados para que seja possível visualizar os dados posteriormente ao encerramento da aplicação. Fazem parte desta entidade os atributos:

Id_teste (numérico): número que identifica cada teste realizado de forma inequívoca. Pode conter no máximo oito algarismos e é incrementado automaticamente.

Data (caracteres): representa a data em que o teste foi efetuado.

Along_i (float): representa o valor do alongamento do cordão em percentagem quando a este é aplicada uma pressão de *i* daN durante um teste. O valor **i** pode assumir os valores 5,10,15,20,25,50,75,100,125,150,175,200,225.

A entidade teste tem relacionamento com:

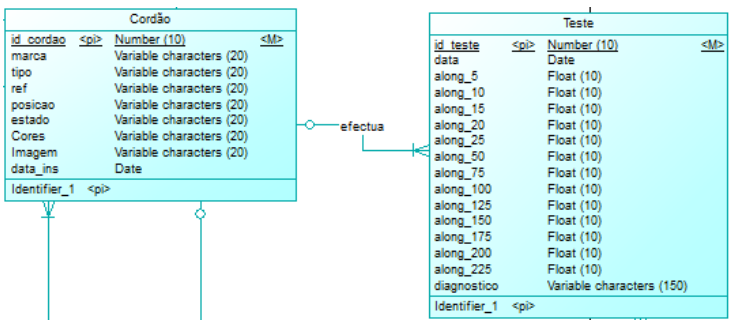
- A entidade cordão com o nome de relação “efetua” com cardinalidade N:1 com participação obrigatória da entidade teste.
- 

Figura 46 – Relacionamento entre a entidade teste e cordao

- A entidade utilizadores com o nome de relação “fazem” com cardinalidade N:1 com participação obrigatória da entidade teste.

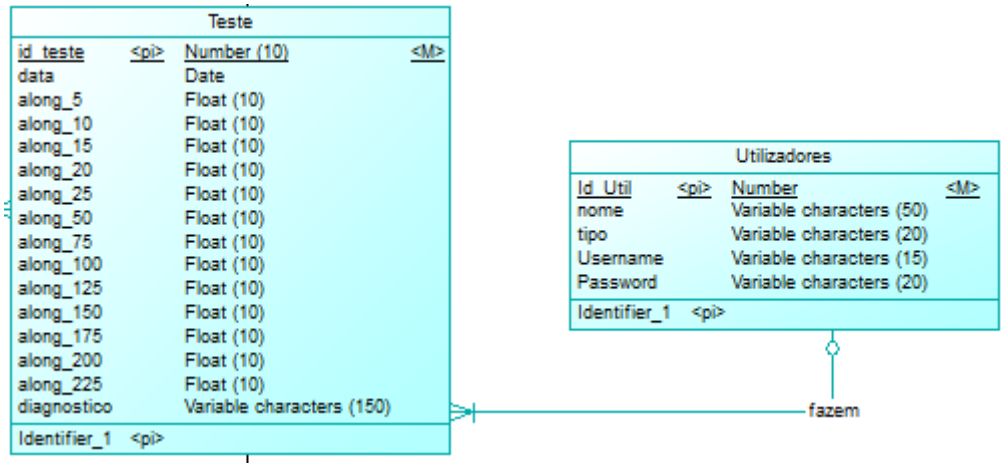


Figura 47 - – Relacionamento entre a entidade teste e utilizadores

Entidade Utilizadores

A entidade utilizadores representa todas as pessoas que têm permissão para aceder à aplicação de testes e consequentemente aceder às funcionalidades da aplicação mediante o tipo de utilizador. Para que possa aceder à aplicação a pessoa terá que constar obrigatoriamente desta entidade. Fazem parte desta entidade os atributos:

Id_util (numérico): número que identifica cada utilizador de forma inequívoca. Pode conter no máximo oito algarismos e é incrementado automaticamente.

Nome(caracteres): contém o nome completo do utilizador com permissões para utilizar a aplicação Sistema de testes.

Tipo(caracteres): representa o tipo de utilizador. Existem vários tipos de utilizadores, nomeadamente" administrador" e "operador" que possuem diferentes permissões no sistema.

Username (caracteres): representa o utilizador que a pessoa que se encontra devidamente registada na base de dados deve utilizar para se autenticar na aplicação.

Password (caracteres): representa a palavra passe que o utilizador deve utilizar para em conjunto com o username efetuar a autenticação na aplicação.

A entidade utilizadores tem relacionamento com:

- A entidade teste com o nome de relação "fazem" com cardinalidade 1:N com obrigatoriedade na entidade teste.

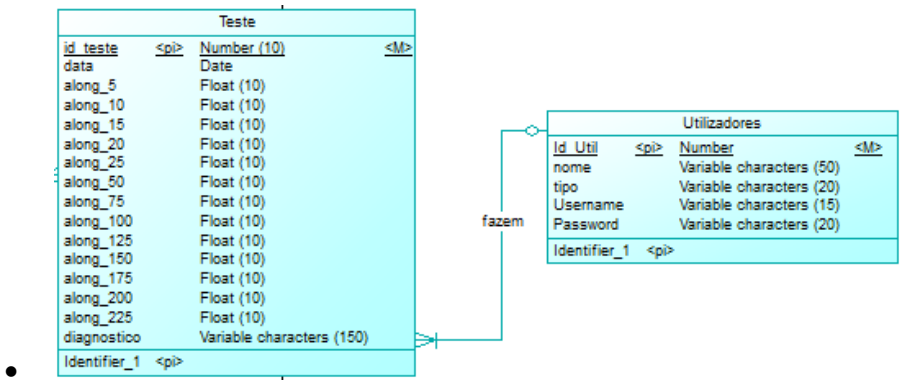


Figura 48 – Relacionamento entre as entidades teste e utilizadores

Entidade DadosFabrica

A entidade dadosfabrica representa os valores de fábrica para determinado cordão. Como foi referido anteriormente, as marcas que produzem cordões divulgam tabelas, como é possível verificar no anexo A, onde é possível verificar vários dados associados a um cordão. A identificação do cordão é dada pela marca juntamente com a referência. Exemplo: Cordão da Marca Liros com a referência LTC45.

Os dados contidos nesta tabela são essenciais para aferir do estado de um cordão que foi submetido a um teste dado que é possível efetuar a comparação entre os valores de fábrica e os valores obtidos no teste.

Fazem parte desta entidade os atributos:

Id_df (numérico): número que identifica cada dado de fábrica de forma inequívoca. Pode conter no máximo oito algarismos e é incrementado automaticamente.

Constituicao (caracteres): representa a constituição de determinada linha/cordão. Como referido anteriormente no capítulo relativo ao estado da arte, o cordão pode ser composto por diversos materiais.

Diâmetro (float): representa o diâmetro de determinado cordão. A unidade usada é o milímetro.

Weight (float): indica o peso associado ao cordão. O resultado é expresso em g/m.

Breakstr (float): indica a carga de tensão ou força mínima necessária para ruptura do cordão. O resultado é expresso em daN.

Effec_bstr (float):): indica a carga de tensão efetiva ou força mínima necessária para ruptura do cordão. O resultado é expresso em daN.

Str_5000cycles (float): Este valor representa a resistência de carga após o cordão ser submetido a um conjunto testes em que estes estão sujeitos a cinco mil curvas de cento e cinquenta graus para cada lado [11].

Along_i (float): representa o valor de fábrica, fornecido pelo fabricante do cordão para o alongamento de um cordão com determinada referência, quando a este é aplicada uma pressão de **i** daN. O valor **i** pode assumir os valores 5,10,15,20,25,50,75,100,125,150,175,200,225. O resultado é expresso em percentagem.

Along_maxstr (float): indica o alongamento máximo do cordão quando lhe é aplicada a tensão que provoca a sua ruptura. O resultado é expresso em percentagem. Exemplo: 3.2%. Este resultado significa que quando é aplicada a tensão de rutura o alongamento do cordão é de mais 3.2% em relação ao seu estado normal.

A entidade dados fabrica tem relacionamento com:

- A entidade dadoscordoes com o nome de relação “origina” com cardinalidade 1:N com participação obrigatória da entidade dadoscordoes.

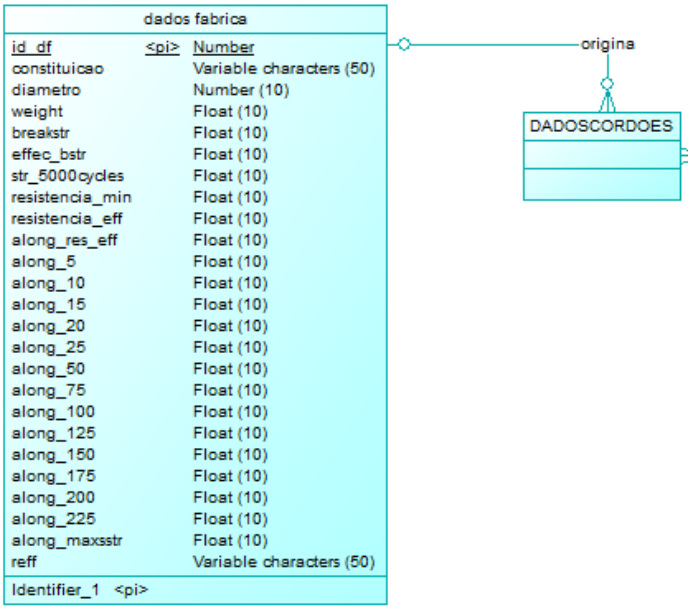


Figura 49 – Relacionamento entre as entidades dados fabrica e dadoscordoes

Entidade ToleranciaCordao

A entidade tolerância_cordao foi criada para armazenar os valores relativos aos pesos que cada tipo de cordão tem. Como referido no capítulo relativo ao estado da arte, um cone de suspensão de parapente entre outros constituintes não menos

importantes, é formado por linhas/cordões. Estas podem ser de tipos diversos mediante a posição que ocupam na asa (A,B,C e D).

Além da avaliação do estado de um cordão o sistema de testes que foi desenvolvido permite aferir o estado de um cone de suspensão do parapente. O utilizador poderá configurar os pesos que pretende dar a cada tipo nesta entidade. Fazem parte desta entidade os atributos:

id_tc (numérico): número que identifica cada registo que é inserido na tabela tolerância_cordao de forma inequívoca. Pode conter no máximo oito algarismos e é incrementado automaticamente.

tipoA (numérico): indica o peso que é atribuído aos cordões do tipo A. Pode assumir os valores {10,20,30,40,50,60,70,80,90,100}.

tipoB (numérico): indica o peso que é atribuído aos cordões do tipo B. Pode assumir os valores {10,20,30,40,50,60,70,80,90,100}.

tipoC (numérico): indica o peso que é atribuído aos cordões do tipo C. Pode assumir os valores {10,20,30,40,50,60,70,80,90,100}.

tipoD (numérico): indica o peso que é atribuído aos cordões do tipo D. Pode assumir os valores {10,20,30,40,50,60,70,80,90,100}.

A entidade tolerância_cordao não possui relacionamentos com nenhuma tabela.

Entidade ResultadoTeste

Esta entidade foi criada para que fosse possível armazenar a classificação de um cone de suspensão de parapente. Fazem parte desta identidade os atributos

Id_rt –(numérico): número que identifica cada registo que é inserido na tabela resultadoteste de forma inequívoca. Pode conter no máximo oito algarismos e é incrementado automaticamente.

Id_asa –(numérico): identifica a que asa/cone de suspensão que pertence cada cordão

Desvio – (numérico): indica o desvio relativo a cada cordão que foi sujeito a um teste.

Entidade ToleranciaMarca

Esta entidade foi criada para que fosse possível configurar pesos diferentes para diferentes marcas. No final de cada teste realizado no sistema de testes, é possível visualizar o resultado que o sistema atribuiu ao estado do cone de suspensão que pode ser: “Muito Bom”,”Bom”,”Suficiente” e “Mau”. O que se pretende é dar a

possibilidade de, por exemplo, para a Marca *Liros,* para se obter um Muito bom o desvio verificado não possa ser maior que 0.1. Por outro lado, pode entender-se que relativamente à marca Cousin esse valor poderá ser de 0.2. Assim,o utilizador tem a liberdade de definir cada valor para os valores Muitos Bom, bom, suficiente e mau relativamente a cada marca. Fazem parte desta entidade os atributos:

Id_tm(numérico): número que identifica cada registo que é inserido na tabela tolerância_marca de forma inequívoca. Pode conter no máximo oito algarismos e é incrementado automaticamente.

Marca (caracteres): indica a marca do cordão. Pode conter no máximo vinte caracteres.

Mb (numérico): se o valor obtido no teste for igual ou menor ao valor contido neste campo a classificação será “Muito Bom”.

B (numérico): se o valor obtido no teste for igual ou menor ao valor contido neste campo e maior que o valor contido no campo mb a classificação será “Muito Bom”.

S (numérico): se o valor obtido no teste for igual ou menor ao valor contido neste campo e maior que o valor contido no campo b a classificação será “Muito Bom”.

M (numérico): se o valor obtido no teste for igual ou maior ao valor contido neste campo a classificação será “Mau”.

A entidade tolerância_marca não possui relacionamentos com nenhuma tabela.

5.3.2 Modelo Entidade Relacionamento (ER)

O modelo Entidade-Relacionamento é um modelo de dados conceptual de alto nível, cujos conceitos foram projetados para estar o mais próximo possível da visão que o utilizador tem dos dados, não se preocupando em representar como estes dados estarão realmente armazenados. [26]

É possível observar na Figura 50 o diagrama de entidade relacionamento relativo à aplicação desenvolvida.

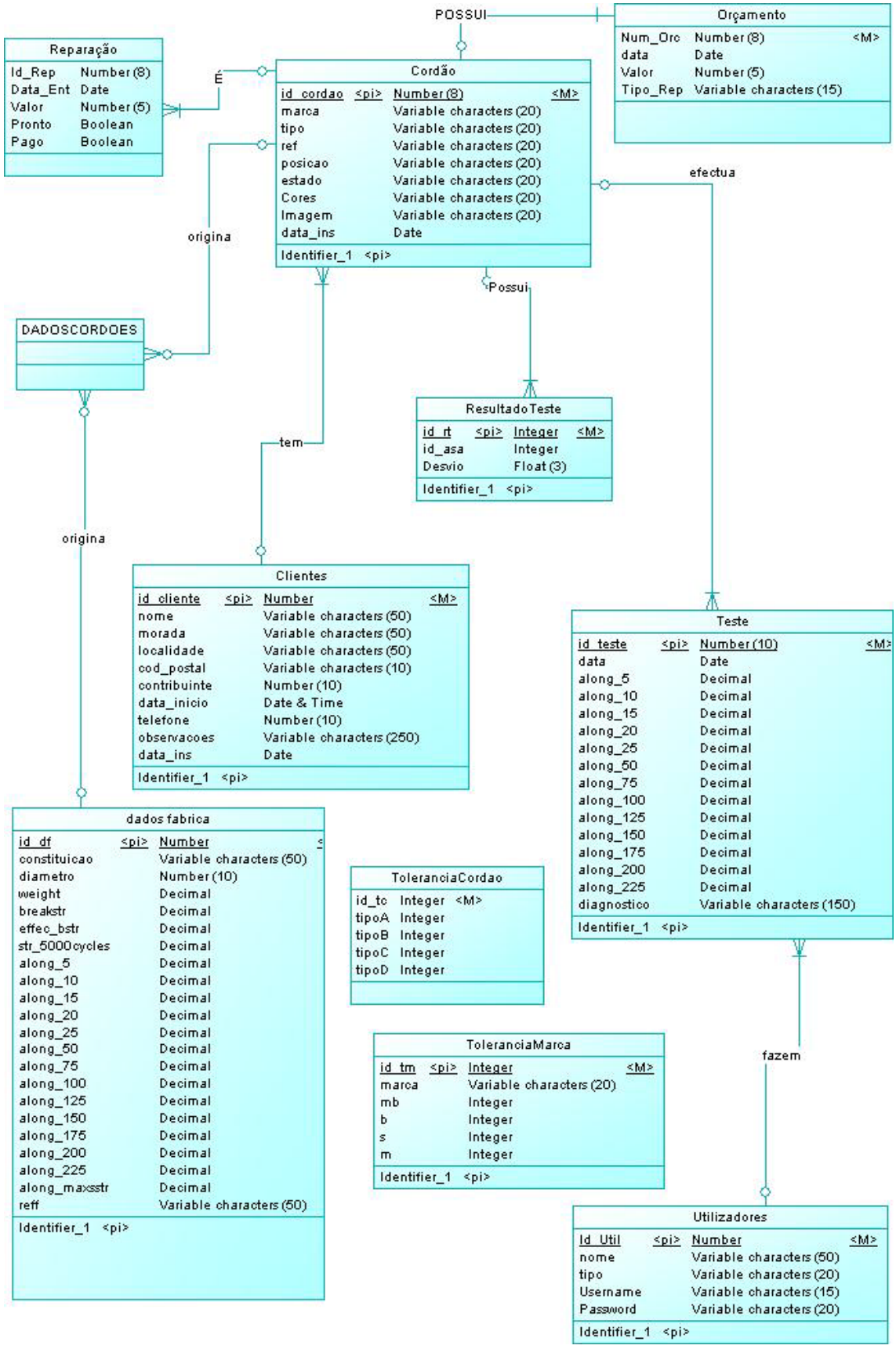


Figura 50 – Diagrama ER

O modelo entidade relacionamento representado na figura 24 foi criado utilizando uma versão *trial* do programa, "*Sybase Power Designer – V15.2*".

O *PowerDesigner* é uma ferramenta poderosa que, entre outras funcionalidades, permite desenhar diagramas de entidade-relacionamento. Fornece todas as vantagens de uma abordagem a dois níveis (a nível conceptual e a nível físico). Além de outras funcionalidades, o *PowerDesigner* permite:

- Modelar um sistema de informação através dos diagramas de entidade relacionamento (CDM - Modelo Conceptual de Dados)

- Gerar o Modelo Físico (PDM – Physical Data Model) correspondente, para um determinado sistema de gestão de Bases de Dados (SGBD).

- Alterar o PDM tendo em conta os parâmetros físicos e considerações de desempenho

- Gerar os scripts SQL para criação das Bases de Dados para o SGBD escolhido.

- A impressão de relatórios do modelo

Após ter sido criado o ER (nível conceptual), foi possível gerar o modelo físico representado Figura 51. No digrama físico que foi gerado, foi necessário verificar se foram geradas todas as tabelas tendo em conta as regras definidas para a criação de diagramas entidade relacionamento. Através da criação do modelo físico foi possível obter o *script* de construção das despectivas tabelas. É possível observar este *script* no anexo B.

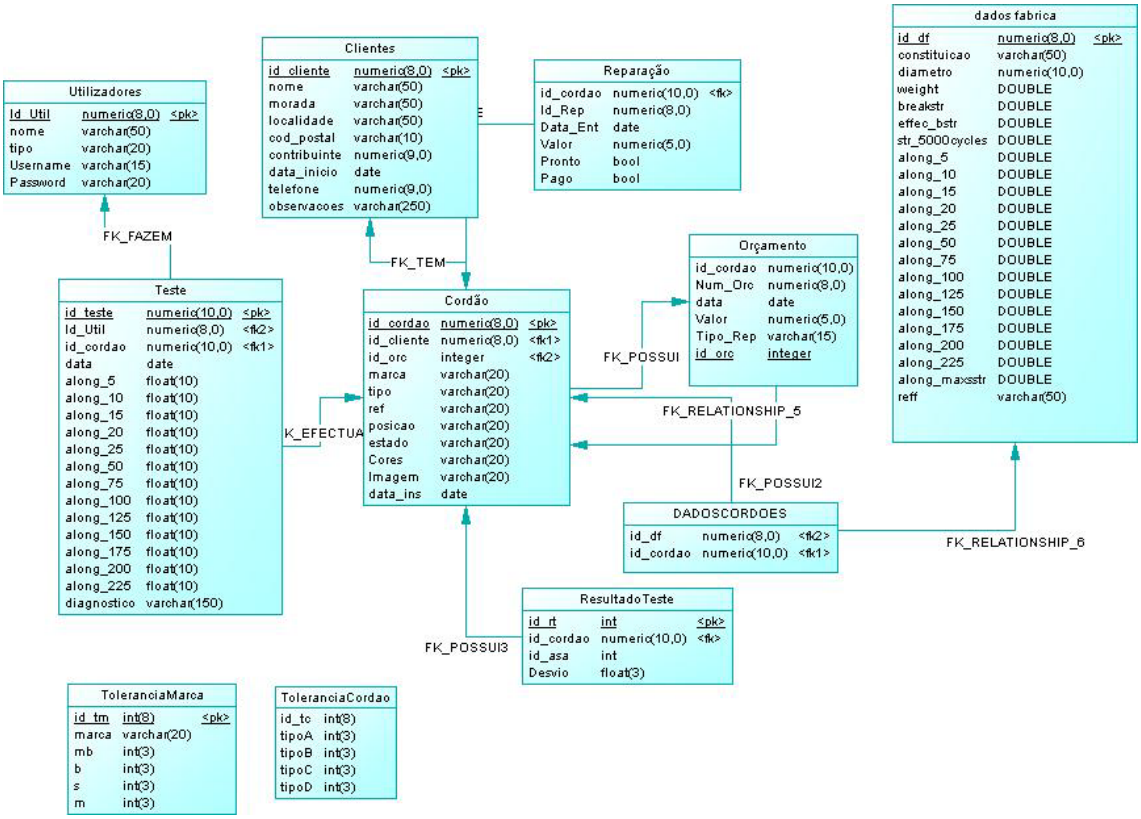


Figura 51 – Modelo Físico

5.4 Subsistema de acesso e codificação da placa de aquisição de dados

Como referido anteriormente, este subsistema tem como uma das responsabilidade estabelecer a comunicação entre a placa de aquisição de dados e o sistema.

O Arduíno UNO, representado na Figura 52, que foi o equipamento escolhido para o desenvolvimento desta aplicação, é capaz de comunicar com um computador através de comunicação serial. Neste caso a ligação ao computador foi efetuada através de cabo *usb*. Esta é vista pelo *software* do computador como uma porta "COM" virtual.

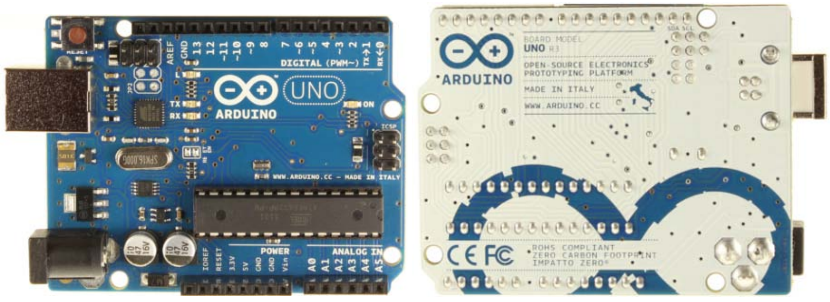


Figura 52 – Arduíno UNO

A classe responsável por estabelecer a comunicação entre o computador e o Arduíno é a classe comunicação. Na Figura 53 apresenta-se um excerto da codificação desta classe.

```
18      class Comunicacao
19      {
20          static SerialPort _serialPort;
21
22          public SerialPort getserialPort()
23          {
24              return _serialPort;
25          }
26          public Comunicacao()
27          {
28              _serialPort = new SerialPort();
29          }
30          public void ConfLigacao(string PortName, int BR)
31          {
32              _serialPort.PortName = PortName;
33              _serialPort.BaudRate = BR;
34          }
35          public bool AbrirPorta()
36          {
37              if (_serialPort.IsOpen == false)
38                  _serialPort.Open();
39              if (_serialPort.IsOpen)
40                  return true;
41              else
42                  return false;
43          }
```

Figura 53 – Excerto do código da classe Comunicacao

Neste excerto de código é possível observar as funções necessárias para o estabelecimento da comunicação entre computador e Arduíno.

Para comunicar, é necessário escrever e ler da porta série. Na Figura 54 é possível observar a função “Escreve” que permite escrever os dados provenientes do Arduíno e a função “Le” que permite ler os dados que são enviados pelo computador e que são dirigidos ao Arduíno.

```
50          public void Escreve(char[] buff)
51          {
52              _serialPort.Write(buff, 0, 1);
53          }
54
55          public string Le()
56          {
57              string receber;
58              receber = _serialPort.ReadExisting();
59              return receber;
60          }
61
```

Figura 54 – Excerto do código da classe Comunicacao (Escrever e Ler)

Do lado do Arduíno, também é preciso implementar as funções que permitem que esta comunicação se efetue. Como referido no capítulo relativo ao Estado da Arte, o Arduíno pode ser programado através de *software* específico (“Arduíno IDE”).

No âmbito desta aplicação os dados que serão enviados dizem respeito aos dados provenientes do sensor de deslocamento e do sensor de pressão. O objetivo é que os dados dos sensores sejam transmitidos em tempo real. Dado que o Arduíno UNO, apenas contém um canal para comunicar, optou-se por enviar os valores dos dois sensores encapsulados numa única variável separados pelo carácter ‘,’. Quando o sistema recebe os valores do Arduíno, consegue separar os dados relativos a cada um dos sensores dado que conhece o elemento separador, que neste caso é uma vírgula.

5.5 Subsistema de criação de gráficos

Dada a importância que a criação de gráficos assume perante o sistema desenvolvido, optou-se por considerar como um subsistema da aplicação.

Para a criação dos gráficos recorreu-se à biblioteca “ZedGraph”. Trata-se de uma biblioteca de utilização livre de acordo com a licença LGPL [29] que contem um conjunto de classes desenvolvidas em C# para criar gráficos de linhas ou barras em 2D. Para se tirar partido desta biblioteca é necessário efetuar um conjunto de parametrizações tendo em conta os dados e a forma como se pretendem representar. A documentação sobre esta ferramenta pode ser encontrada em [54].

É possível visualizar os gráficos relativos a testes já efetuados e que se encontram guardados na base de dados, assim como no momento em que é realizado um teste a um cordão.

Durante um teste a um cordão são exibidos vários tipos de gráficos. Existe um tipo que projeta os valores recebidos pelo sensor de pressão em função do tempo e que pode ser observado na Figura 55. Outro representa os dados provenientes do

sensor de deslocamento em função do tempo que corresponde à Figura 56. Existe também um gráfico que efetua uma representação dos dados referentes à pressão e ao deslocamento que pode ser observado na Figura 57.

Os gráficos são atualizados de segundo a segundo com os valores recebidos pelos sensores. Neste caso, como os sensores estão a medir valores relativos à temperatura, alguns gráficos não refletem o comportamento esperado para o teste. É o caso do gráfico que expressa a pressão em função do tempo (Figura 55). O resultado esperado para o gráfico deveria compreender valores para a pressão entre 5 e 225. O comportamento da curva também não é o esperado porque em alguns momentos o valor para a pressão é inferior relativamente a tempos anteriores.

Como já referido anteriormente, ainda não existe o material adequado, nomeadamente neste caso o sensor de carga. A configuração seria idêntica, bastando substituir no sistema o sensor de temperatura pelo sensor de carga.

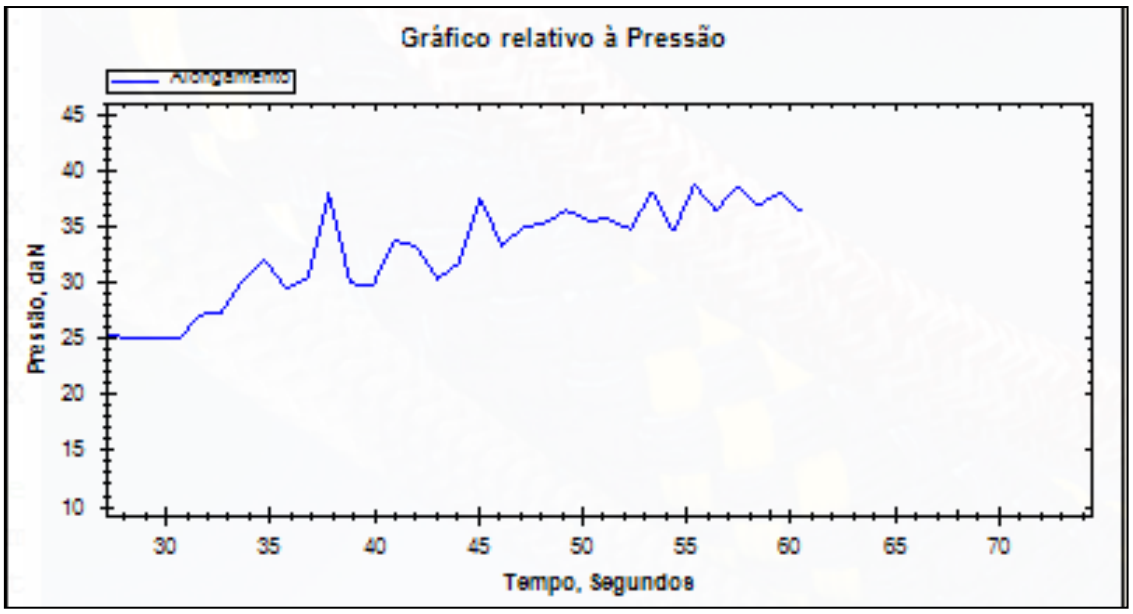


Figura 55 – Gráfico Pressão/Tempo

No caso do gráfico que expressas o alongamento de um cordão em função do tempo o comportamento da curva é o esperado. Ao longo de um teste, em virtude da carga que é aplicada a um cordão, o seu alongamento vai aumentar.

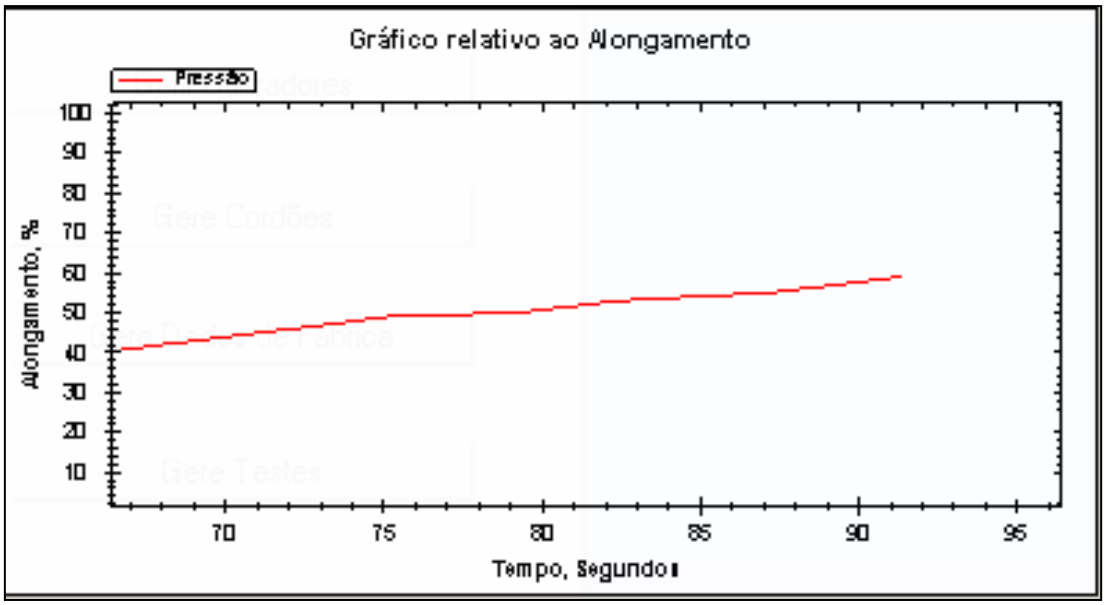


Figura 56 – Gráfico Alongamento/Tempo

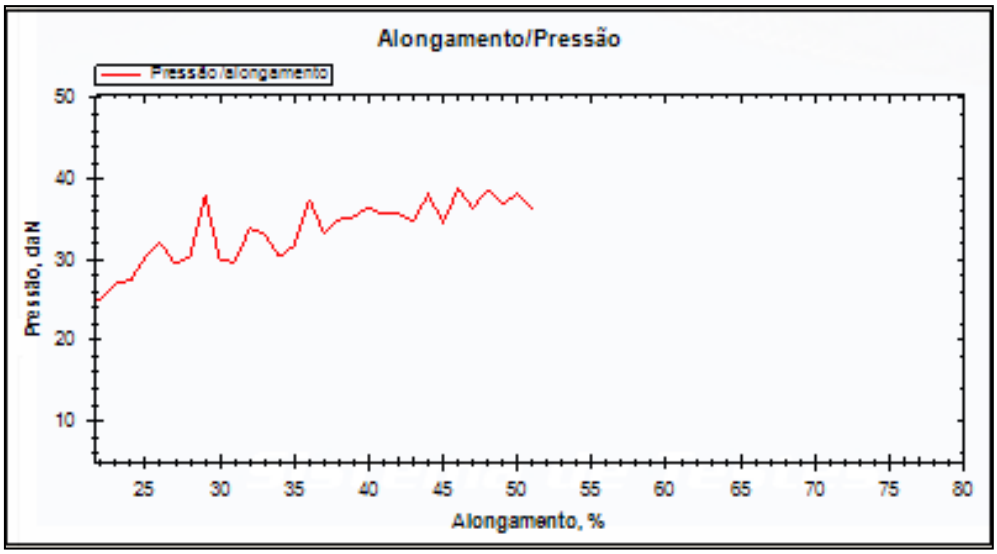


Figura 57 – Gráfico Pressão/Deslocamento

No final é exibido um gráfico (Figura 58) onde é possível observar a comparação entre os dados obtidos durante o teste e os dados de fábrica. No caso deste gráfico, no que diz respeito à linha vermelha, que representa os dados de fábrica, verifica-se que o resultado é o esperado. Isto acontece porque estes valores dizem respeito a dados que previamente foram inseridos na base de dados e que dizem respeito aos dados de fábrica dos cordões.

Os dados de fábrica, como referido anteriormente, dizem respeito aos dados que são divulgados pelas marcas e que representam os valores ideais para determinado cordão. O cordão é identificado através da sua marca e referência pelo que é possível obter na tabela ”DadosFabrica” os dados relativos ao cordão que se testou.

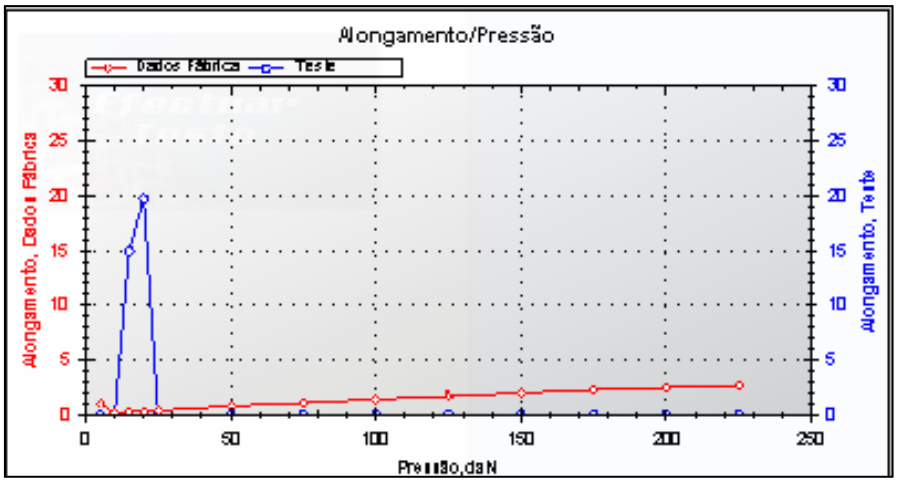


Figura 58 – Gráfico Dados Fábrica/Teste

5.6 Síntese

No presente capítulo, para além da apresentação da aplicação desenvolvida, foram especificados os vários subsistemas em que foi dividido o sistema:

- Subsistema de Gestão da Aplicação
- Subsistema de acesso ao SGBD
- Subsistema de acesso e codificação da placa de aquisição de dados
- Subsistema de criação de gráficos

Capítulo 6

Conclusões e Trabalho Futuro

As conclusões mais relevantes do trabalho desenvolvido que conduziram a elaboração desta dissertação, assim como as frentes abertas para trabalho futuro, serão a seguir apresentadas.

6.1 Conclusões

A ideia para desenvolver um projeto deste tipo surgiu após verificar-se que em Portugal os únicos testes que são efetuados aos cordões de parapente são meramente visuais. Este tipo de testes não é fiável e reduz consideravelmente o grau de segurança dos pilotos de parapente.

Atendendo à conjetura económica atual, é essencial que as empresas apostem cada vez mais na diferenciação dos seus serviços tendo em conta alcançar novas vantagens competitivas. O protótipo desenvolvido durante a dissertação de mestrado pretende no futuro introduzir novas competências nas empresas especializadas em reparações de asas de parapente.

O projeto apresentado consistiu na construção de um protótipo que permitisse analisar o estado dos cordões que formam um cone de suspensão de parapente. Durante o seu desenvolvimento foram implementadas várias funcionalidades:

- Capacidade de efetuar cálculos relativamente ao estado de um cordão, como o desvio médio ;
- Classificação de um cone de suspensão quanto ao seu estado de conservação;
- Possibilidade de visualizar, em tempo real, gráficos relativos aos testes efetuados aos cordões;

- Gestão de diversas entidades: cordões, testes, dados de fábrica, clientes, utilizadores.
- Possibilidade de alterar os pesos atribuídos aos vários grupos de cordões do parapente (A, B, C e D);
- Possibilidade de definir os valores para as diferentes marcas e que servirão de base para a classificação do estado do cone de suspensão.

Dado que o material necessário para o desenvolvimento da aplicação não foi adquirido atempadamente pela entidade colaborante, foi necessário pensar em alternativas. Embora não fosse essencial a existência de equipamento, até porque não fazia parte dos objetivos assumidos para a realização desta dissertação a execução de teste reais a cordões de parapente, entendeu-se que a respetiva aquisição facilitaria o desenvolvimento. Para colmatar este problema foi adquirido um kit Arduíno com um conjunto de sensores. Apesar do equipamento diferir do escolhido para o projeto, possibilitou que fossem gerados dados reais e também que fosse verificada a validade dos métodos desenvolvidos.

O projeto desenvolvido incluiu a colaboração de uma empresa especializada na reparação de parapentes, a *ParaClinic.* Entende-se que este tipo de parceria vem de encontro à vertente profissionalizante que é pretendida para este tipo de projetos.

6.2 Melhoramentos Futuros

Apesar de na generalidade terem sido atingidos os objetivos delineados para esta dissertação, será importante no futuro submeter o protótipo a testes reais. Esta situação encontra-se pendente da aquisição de todo o material necessário pela empresa colaborante.

O projeto desenvolvido diz respeito a um protótipo, que tal como o nome indica não é uma solução final. Uma vez que a aplicação foi construída respeitando a programação orientada a objetos, será facilmente alterada caso seja necessário.

A médio prazo, apontam-se alguns novos requisitos agrupados em funcionais e não funcionais.

Requisitos Funcionais

- Incluir nos sistemas de testes a possibilidade de interagir com um maior número de sistemas de aquisição de dados

- Possibilitar a introdução de imagens relativas aos cordões
- Proteger todos os campos de inserção de dados relativamente aos dados introduzidos pelo utilizador.

Requisitos Não Funcionais

- Melhorar significativamente o aspeto gráfico da aplicação
- Dado que se trata de um protótipo, não foi dedicada especial atenção ao capítulo da usabilidade. Pretende-se rever a aplicação para verificar se poderão ser efetuadas alterações que melhorem a aplicação neste sentido

Referências Bibliográficas

[1] A flexible charting library for .NET - CodeProject. Obtido Dezembro 8, 2011, de http://www.codeproject.com/KB/graphics/zedgraph.aspx

[2] Aguiar, G.. Mapeando dependências entre tabelas. Obtido Agosto 21, 2011, de http://gustavomaiaaguiar.wordpress.com/2008/09/12/mapeando-dependencias-entre-tabelas/

[3] Ahmed, Mesbah; GARRET, Chris, FAIRCLOTH, Jeremy, PAYNE, Chris[2002],ASP:NET Web Developer´s Guide, Syngress, 2002

[4] Algarve e:Motion - Portal de desportos radicais e alternativos no Algarve :: Ar. Obtido Dezembro 8, 2011, b de http://www.algarvemotion.com/modalidade-ar.php?id=15

[5] America's #1 Powered Paraglider & Paramotor manufacturer, Out selling all others by more then 10 to 1, ParaToys & BlackHawk.Obtido Dezembro 8, 2011, de http://paratoys.com/

[6] Arduino - ArduinoBoardUno.Obtido Dezembro 8, 2011, de http://www.arduino.cc/en/Main/ArduinoBoardUno

[7] Arduino – Wikipédia, a enciclopédia livre.Obtido Dezembro 8, 2011, de http://pt.wikipedia.org/wiki/Arduino

[8] Arduino Brasil: Arduino + Sensor Temperatura + Visual c# parte 1/2.Obtido Dezembro 8, 2011, de http://arduinobrasil.blogspot.com/2010/02/arduino-sensor-temperatura-visual-c.html

[9] Arduino Uno | Multilógica-shop.Obtido Dezembro 8, 2011, de http://multilogica-shop.com/Arduino-Uno

[10] ArrayList: Dynamic array in .NET. Obtido Dezembro 8, 2011, de http://www.csharpfriends.com/Articles/getArticle.aspx?articleID=56

[11] Article: Paraglider Lines / www.ojovolador.com.Obtido Maio 20, 2011, de http://www.ojovolador.com/eng/read/reports/lines/index.htm

[12] Asas São Miguel - Boletim IV.Obtido Junho 1, 2011, de http://www.asassaomiguel.com/index.php?option=com_content&task=view&id=672&Itemid=50

[13] AVLP - Associação de Vôo Livre de Pancas. Obtido de http://www.avlp.com.br/site/

[14] Banzi, M. (2009). Getting started with Arduino (1st ed.). Beijing ;;Cambridge: Make Books / O'Reilly.

[15] Bendat, J. (1986). Random data : analysis and measurement procedures (2nd ed.). New York: Wiley.

[16] Brawley, P., & Fuller, A. (2010). Get It Done With MySQL 5, Chapter 1. Obtido Fevereiro 22, 2011, de http://www.artfulsoftware.com/mysqlbook/sampler/mysqled1ch01.html

[17] Breaking Strength. Obtido Agosto 8, 2011, de http://www.instron.com/wa/glossary/Breaking-Strength.aspx

[18] Chand, M..Timer in C#. Obtido Junho 5, 2011, de http://www.c-sharpcorner.com/UploadFile/mahesh/WorkingwithTimerControlinCSharp11302005054911AM/WorkingwithTimerControlinCSharp.aspx

[19] Change Vision — Astah Community, UML, Professional, Share and iPad.Obtido Outubro 2, 2011, de http://astah.net/

[20] Cousin Trestec | référence technique et professionnelle du cordage.Obtido Dezembro 9, 2011, de http://www.cousin-trestec.com/

[21] Cunningham, W., Enterprise Solution Patterns Using Microsoft .NET, Microsoft Press, 2003

[22] Damas, L. SQL - Structured Query Language (5th ed.). FCA.

[23] Delgado, A. Make Bits - Tudo sobre Robótica e Arduino em Portugal. Obtido Maio 18, 2011, de http://makebits.net/

[24] Desenvolvimento em Camadas. Obtido Março 20, 2011, de http://www.microsoft.com/brasil/msdn/tecnologias/arquitetura/Layers_Developing.mspx

[25] DHV Hanggliding and Paragliding in Germany: Paragliding and Hanggliding Equipment Databases.Obtido Maio 19, 2011, de http://www.dhv.de/web/index.php?id=651

[26] Diagrama Entidade Relacionamento. Obtido Dezembro 8, 2011, de http://pt.scribd.com/doc/50184885/31/Diagrama-Entidade-Relacionamento

[27] Dobson, Rick [2002], Programming Microsoft SQL Server 2000 with Microsoft Visual Basic.NET, Microsoft Corporation, 2002

[28] Federação Portuguesa de Aeronáutica Obtido Dezembro 9, 2011, de http://www.fpaero.pt/comissao_tecnica.php?comissao_id=6

[29] GNU Lesser General Public License - Wikipedia, the free encyclopedia.Obtido Outubro 8, 2011, de http://en.wikipedia.org/wiki/LGPL

[30] Guia: Como Comprar um Paraglider - MercadoLivre.Obtido Abril 20, 2010, de http://guia.mercadolivre.com.br/como-comprar-paraglider-10628-VGP

[31] Inspections | About Paragliders | Dudek Paragliders.).Obtido Abril 2, 2011, de http://www.dudek.eu/en/inspections/

[32] John Park, & Steve Mackay. (2003). Practical data acquisition for instrumentation and control systems. Oxford ;;Boston: Elsevier.

[33] Linear Potentiometer, Position Transducer, Displacement Sensors, String Pot. Obtido Janeiro 8, 2011, de http://www.unimeasure.com/std-a.htm

[34] Macorati, João Carlos, UML – Diagrama de Classes e Objectos, [online]. Disponível na Internet via WWW. URL: http://www.macoratti.net/net_uml1.htm [4]

[35] Marques, P., Hernâni, P., & Figueira, R. (2009). C# 3.5. FCA.

[36] McRoberts, M. (2009, Maio). Arduino Starter Kit Manual A Complete Beginners Guide to the Arduino. hshine Electronics. Obtido de www.earthshineelectronics.com

[37] Microsoft [2002b], Application Architecture for .NET: Designing Applications and Services, Microsoft Corporation, 2002

[38] My Projects: Arduino LM35 Sensor. Obtido Julho 8, 2011, de http://pscmpf.blogspot.com/2008/12/arduino-lm35-sensor.html

[39] MySQL – Disable Foreign Key Checks or Constraints | Me and My Thoughts. (sem data). Obtido Outubro 4, 2011, de http://gauravsohoni.wordpress.com/2009/03/09/mysql-disable-foreign-key-checks-or-constraints/

[40] MySQL – Wikipédia, a enciclopédia livre. Obtido Abril 28, 2011, de http://pt.wikipedia.org/wiki/MySQL

[41] Nagel, C. (2010). Professional C# 4 and .Net 4. Indianapolis IN: Wiley Pub.

[42] Nunes, Mauro e O'NEILL Henrique, Fundamental de UML, FCA – Editora de Informática, Dezembro 2004.

[43] O Parapente Cross Country | Vertigens. Obtido Abril 19, 2010, de http://vertigens.com/artigos/parapente-cross-country

[44] Pagen, D. (2001). The art of paragliding : learning paragliding skills for beginner to intermediate pilots. Spring Mills Pa.: D. Pagen.

[45] Parapente Sul. Obtido Maio 19, 2011, de http://www.parapentesul.com.br/conteudo.php?id=9

[46] Pereira, A., & Poupa, C. (2008). Como escrever uma Tese. Lisboa: Edições Sílabo.

[47] Puntar, Sérgio, Métodos e Visualização de Grupamentos de Dados, Tese de Mestrado, COPPE/UFRJ, Julho de 2003

[48] S-Bean, Load Cell All Stainless Steel Construction, High Accuracy, Economical Price. Obtido Setembro 25, 2011, de http://www.omega.com/pptst/LC101.html

[49] SOL Paragliders – Manutenção. Obtido Maio 19, 2011, de http://www.solparagliders.com.br/br/content/blogcategory/46/53/

[50] Spicer, J. (2009, Setembro 13). Connect to the Arduino with C#. Obtido Maio 15, 2011, de http://www.technicana.com/physical-computing/73-connect-to-the-arduino-with-c-

[51] Support Forum (EN/FR) • Search. Obtido Fevereiro 3, 2011, de http://forum.velleman.eu/search.php?keywords=vm140&terms=all&author=&sc=1&sf=all&sk=t&sd=d&sr=posts&st=0&ch=300&t=0&submit=Search

[52] Velleman nv. Obtido Setembro 9, 2011, de http://www.velleman.eu/

[53] Wind ::: Newsletter 64 - Informação Técnica – DHV e EN. Obtido Junho 7, 2011, de http://www.sam-cam.com/newsletter/news64.html

[54] ZedGraph | Free Science & Engineering software downloads at SourceForge.net. Obtido Janeiro 9, 2011, de http://sourceforge.net/projects/zedgraph/

[55] ZM Oficina de Parapente. Obtido Maio 19, 2011, de http://www.zmoficinadeparapente.com.br/index.php?id=81

Anexos

Anexo A – Tabelas relativas a dados de fábricas de marcas

COUSIN TRESTEC

Lines for PARAGLIDERS

Lignes DYNEEMA® Ultimate

SAMPLES season 2007

Dyneema pre-stretched braid with polyester cover. Lighter and more resistant.

Sample	I	J	K	L	M	N	O	P
Reference	**0.95mm**	**1mm**	**1.1mm**	**1.3mm**	**1.4mm**	**1.5mm**	**1.9mm**	**2.1mm**
Material sleeve	Polyester	Polyester	Polyester	Polyester	Polyester	Polyester	Polyester	Polyester
Material core	Dynnema	Dyneema	Dyneema	Dyneema	Dyneema	Dyneema	Dyneema	Dyneema
Type core	Braided	Twisted	Twisted	Braided	Braided	Braided	Braided	Braided
Diameter in mm	0.95	1	1.1	1.3	1.4	1.5	1.9	2.1
Weight per meter total g/m	0.78	0.78	1	1.21	1.6	1.78	2.41	3.3
Weight per meter core g/m	0.27	0.3	0.37	0.55	0.54	0.74	1.07	1.46
Weight per meter sleeve g/m	0.51	0.48	0.63	0.66	1.06	1.04	1.34	1.84
Breaking strength mini daN	80	100	115	130	160	200	310	380
Effective breaking strength daN	**86**	**110**	**128**	**146**	**165**	**216**	**328**	**413**
Strength after 5000 bending cycles daN	75	94	105	119	145	198	292	380
Elongation at 5 daN (%)	0.1	0.1	0.1	0.05	0.05	0.05	0	0.05
Elongation at 10 daN (%)	0.2	0.2	0.2	0.2	0.15	0.1	0.05	0.1
Elongation at 15 daN (%)	0.4	0.4	0.4	0.3	0.3	0.2	0.1	0.15
Elongation at 20 daN (%)	0.6	0.55	0.5	0.5	0.35	0.3	0.15	0.15
Elongation at 25 daN (%)	0.7	0.7	0.7	0.6	0.55	0.4	0.2	0.3
Elongation at 50 daN (%)	1.7	1.3	1.2	1.2	0.95	0.6	0.4	0.35
Elongation at 75 daN (%)	2.55	2.1	2.0	1.8	1.35	1.0	0.55	0.5
Elongation at 100 daN (%)		2.5	2.6	2.4	1.75	1.3	0.8	0.7
Elongation at 125 daN (%)				3	2.4	1.6	0.95	0.85
Elongation at 150 daN (%)					2.9	2.1	1.2	1.0
Elongation at 175 daN (%)						2.45	1.5	1.2
Elongation at 200 daN (%)						2.8	1.75	1.45
Elongation at 225 daN (%)							2.2	1.55
Elongation at breaking strength (%)	2.55	2.5	2.8	3.1	3.0	2.8	3.0	2.6

Q Polyester braid Ref 918 - Diameter mm: 2.14 - Weight/m: 3.36 g/m - BS average: 90 daN

RS Technora webbing Ref 3455 - Width: 12 mm - Weight/m: 9.1 g/m - BS average: 1078 daN

T Nylon webbing Ref 608 - Width: 10 mm - Weight/m : 5.6 g/m - RR : 200 daN

Date : 22/03/2007

COUSIN TRESTEC LABORATORY

These values were obtained on new braids during different controls in our laboratory. We remind that the execution with sewn eye or knot change these values.

Non-contractual values, we reserve the right for changes of the technical characteristics of presented products, without notice.

1daN = ± 1,02 kg = ± 2,2 lbs. Dyneema ® : DSM registered trade mark - Technora ® : Teijin-Twaron registered trade mark.

COUSIN TRESTEC - 8, rue Abbé Bonpain - BP 70020 Wervicq sud - 59558 Comines Cédex - France - Phone:+33 3.20.14.41.35 - Fax:+33 3.20.39.59.12

C4/1/07 version 3

LIROS
FLIGHTSPORTTECHNIQUE

ROSENBERGER
TAUWERK GMBH
Poststr.11
D-95192 Lichtenberg
tel.: +49 9288 71-32
fax: +49 9288 71-24
mail: fg@liros.com
www.liros.com

LIROS LTC

Technical data

NEW

Ungemantelte Gleitschirmleinen, Technora + ARC-Coating
Uncovered paraglider lines, Technora + ARC-Coating

	Durch-messer Dia-meter [mm]	Bruchlast min. Strength min. [daN]	Bruchlast eff. Strength eff. [daN]	Bruch-dehnung Elongation at Strength eff. %	Dehnung / Kriechen bei 10 daN Elongation /Line creep at 10 daN %	Dehnung / Kriechen bei 30 daN Elongation / Line creep at 30 daN %	Festigkeit nach Knicktest Strength after 5000 bending cycles [daN]	Gesamt-masse Weight [g/m]
LTC45	0,55	45	58	5,2	1,20	2,60	37	0,31
LTC65	0,65	65	79	5,1	0,85	2,05	43	0,51
LTC80	0,70	80	103	4,6	0,60	1,50	76	0,67
LTC120	1,10	120	172	3,5	0,45	1,05	96	0,88
LTC160	1,20	160	201	4,5	0,60	1,15	135	1,13
LTC200	1,30	200	244	4,7	0,65	1,15	149	1,39

Anexo B – Atributos das Tabelas

Tabela 4 – Atributos da entidade cordao

Nome do atributo	Aceita nulos?	Valores únicos?	Observações
Id_cordão	N	S	Identificador (chave primaria), não admite nulos. Apenas existe um número para cada cordão, nunca existe um número para dois cordões. O número máximo de dígitos é oito.
Marca	S	N	Valor que admite nulos e poderá existir a mesma marca para vários cordões. O número de caracteres máximo é vinte.
Tipo	S	N	Valor que admite nulos e poderá existir o mesmo tipo para vários cordões. O número de caracteres máximo é vinte.
Ref	S	N	Valor que admite nulos e poderá existir a mesma referência para vários cordões. O número de caracteres máximo é vinte.
Posicao	S	N	Valor que admite nulos e poderá existir a mesma posição para vários cordões. O número de caracteres máximo é vinte.
estado	S	N	Valor que admite nulos e poderá existir o mesmo estado para vários cordões. O número de caracteres máximo é vinte.
cores	S	N	Valor que admite nulos e poderá existir as mesmas cores para vários cordões. O número de caracteres máximo é vinte.
imagem	S	S	Valor que admite nulos. O número máximo de caracteres é vinte.

Tabela 5 – Atributos da entidade cliente

Nome do atributo	Aceita nulos?	Valores únicos?	Observações
Id_cliente	N	S	Identificador (chave primaria), não admite nulos. Apenas existe um id_cliente para cada cliente.
Nome	N	N	Campo que não pode ser nulo. Podem existir nomes iguais que serão diferenciados através do id_cliente.
morada	S	N	Campo que pode ser nulo e que admite valores repetidos.
localidade	S	N	Campo que pode ser nulo e que admite valores repetidos.
Cod_postal	S	N	Campo que pode ser nulo e que admite valores repetidos.
contribuinte	S	N	Campo que pode ser nulo e que admite valores repetidos.
telefone	S	N	Campo que pode ser nulo e que admite valores repetidos.
observacoes	S	N	Campo que pode ser nulo e que admite valores repetidos.

Tabela 6 – Atributos da entidade teste

Nome do atributo	Aceita nulos?	Valores únicos?	Observações
Id_teste	N	S	Identificador (chave primaria), não admite nulos. Apenas existe um numero para cada teste.
data	N	N	Valor que representa a data em que foi efectuado o teste.
Along_5	S	N	Admite valores nulos.
Along_10	S	N	Admite valores nulos.
Along_15	S	N	Admite valores nulos.
Along_20	S	N	Admite valores nulos.
Along_25	S	N	Admite valores nulos.
Along_50	S	N	Admite valores nulos.
Along_75	S	N	Admite valores nulos.
Along_100	S	N	Admite valores nulos.
Along_125	S	N	Admite valores nulos.
Along_150	S	N	Admite valores nulos.
Along_175	S	N	Admite valores nulos.
Along_200	S	N	Admite valores nulos.
Along_225	S	N	Admite valores nulos.

Tabela 7 – Atributos da entidade utilizador

Nome do atributo	Aceita nulos?	Valores únicos?	Observações
Id_util	N	S	Identificador (chave primaria), não admite nulos. Apenas existe um numero para cada utilizador.
nome	N	N	Valor não nulo. Indica o nome do utilizador.
tipo	N	N	Valor não nulo. Indica o tipo de utilizador.
username	N	S	Não admite nulos. Não podem existir dois registos com username iguais.
password	N	N	Não admite nulos.

Tabela 8 – atributos da entidade dados fabrica

Nome do atributo	Aceita nulos?	Valores únicos?	Observações
Id_df	N	S	Identificador (chave primaria), não admite nulos. Apenas existe um numero para identificar cada dado de fábrica que é inserido na base de dados.
Constituicao	S	N	Valor que pode ser.
Diâmetro	S	N	Valor não nulo. Indica o tipo de utilizador.
Weight	S	S	Não admite nulos. Não podem existir dois registos com username iguais.
Breakstr	S	N	Admite valores nulos.
Effec_bstr	S	N	Admite valores nulos.
Str_5000cycles	S	N	Admite valores nulos.
Along_5	N	N	Não admite nulos.
Along_10	N	N	Não admite nulos.
Along_15	S	N	Admite valores nulos.
Along_20	S	N	Admite valores nulos.
Along_25	S	N	Admite valores nulos.
Along_50	S	N	Admite valores nulos.
Along_75	S	N	Admite valores nulos.
Along_100	S	N	Admite valores nulos.
Along_125	S	N	Admite valores nulos.
Along_150	S	N	Admite valores nulos.
Along_175	S	N	Admite valores nulos.
Along_200	S	N	Admite valores nulos.
Along_225	S	N	Admite valores nulos.
Along_maxstr	S	N	Admite valores nulos.

Tabela 9 – Atributos da entidade tolerância_cordao

Nome do atributo	Aceita nulos?	Valores únicos?	Observações
Id_tc	N	S	Identificador (chave primaria), não admite nulos. Apenas existe um numero para identificar cada registo desta tabela que é inserido na base de dados.
tipoA	N	S	Não admite nulos. Pode assumir os valores {10,20,30,40,50,60,70,80,90,100}
tipoB	N	S	Não admite nulos. Pode assumir os valores {10,20,30,40,50,60,70,80,90,100}
tipoC	N	S	Não admite nulos. Pode assumir os valores {10,20,30,40,50,60,70,80,90,100}
tipoD	N	S	Não admite nulos. Pode assumir os valores {10,20,30,40,50,60,70,80,90,100}

Tabela 10 – Atributos da tabela tolerância_marca

Nome do atributo	Aceita nulos?	Valores únicos?	Observações
Id_tm	N	S	Identificador (chave primaria), não admite nulos. Apenas existe um numero para identificar cada registo desta tabela que é inserido na base de dados.
marca	N	N	Identifica a marca do cordão. Pode ter no máximo vinte caracteres.
mb	N	N	Pode assumir os valores {5,10,15,20,25,30,35,40,45,50,55,60,65,70,75,80,85,90,95,100}
b	N	N	Pode assumir os valores {5,10,15,20,25,30,35,40,45,50,55,60,65,70,75,80,85,90,95,100}
s	N	N	Pode assumir os valores {5,10,15,20,25,30,35,40,45,50,55,60,65,70,75,80,85,90,95,100}
m	N	N	Pode assumir os valores {5,10,15,20,25,30,35,40,45,50,55,60,65,70,75,80,85,90,95,100}

Anexo C – Script de criação da Base de Dados

```
/*==============================================================
*/
/* DBMS name:      MySQL 5.0
*/
/* Created on:     27-11-2011 19:47:57
*/
/*==============================================================
*/


drop table if exists CLIENTES;

drop table if exists CORDAO;

drop table if exists DADOSCORDOES;

drop table if exists DADOS_FABRICA;

drop table if exists ORCAMENTO;

drop table if exists REPARACAO;

drop table if exists RESULTADOTESTE;

drop table if exists TESTE;

drop table if exists TOLERANCIACORDAO;

drop table if exists TOLERANCIAMARCA;

drop table if exists UTILIZADORES;

/*==============================================================
*/
/* Table: CLIENTES
*/
/*==============================================================
*/
create table CLIENTES
(
   ID_CLIENTE           numeric(8,0) not null,
   NOME                 varchar(50),
   MORADA               varchar(50),
   LOCALIDADE           varchar(50),
   COD_POSTAL           varchar(10),
   CONTRIBUINTE         numeric(9,0),
   DATA_INICIO          date,
   TELEFONE             numeric(9,0),
   OBSERVACOES          varchar(250),
   primary key (ID_CLIENTE))
type = InnoDB;
```

```
/*==============================================================
*/
/* Table: CORDAO
*/
/*==============================================================
*/
create table CORDAO
(
   ID_CORDAO            numeric(8,0) not null,
   ID_CLIENTE           numeric(8,0),
   ID_ORC               integer,
   MARCA                varchar(20),
   TIPO                 varchar(20),
   REF                  varchar(20),
   POSICAO              varchar(20),
   ESTADO               varchar(20),
   CORES                varchar(20),
   IMAGEM               varchar(20),
   DATA_INS             date,
   primary key (ID_CORDAO)
)
type = InnoDB;

/*==============================================================
*/
/* Table: DADOSCORDOES
*/
/*==============================================================
*/
create table DADOSCORDOES
(
   ID_DF                numeric(8,0),
   ID_CORDAO            numeric(10,0)
)
type = InnoDB;

/*==============================================================
*/
/* Table: DADOS_FABRICA
*/
/*==============================================================
*/
create table DADOS_FABRICA
(
   ID_DF                numeric(8,0) not null,
   CONSTITUICAO         varchar(50),
   DIAMETRO             numeric(10,0),
   WEIGHT               DOUBLE,
   BREAKSTR             DOUBLE,
   EFFEC_BSTR           DOUBLE,
   STR_5000CYCLES       DOUBLE,
   ALONG_5              DOUBLE,
   ALONG_10             DOUBLE,
   ALONG_15             DOUBLE,
   ALONG_20             DOUBLE,
   ALONG_25             DOUBLE,
```

```
   ALONG_50             DOUBLE,
   ALONG_75             DOUBLE,
   ALONG_100            DOUBLE,
   ALONG_125            DOUBLE,
   ALONG_150            DOUBLE,
   ALONG_175            DOUBLE,
   ALONG_200            DOUBLE,
   ALONG_225            DOUBLE,
   ALONG_MAXSSTR        DOUBLE,
   REFF                 varchar(50),
   primary key (ID_DF)
)
type = InnoDB;

/*==============================================================
*/
/* Table: ORCAMENTO
*/
/*==============================================================
*/
create table ORCAMENTO
(
   ID_CORDAO            numeric(10,0),
   NUM_ORC              numeric(8,0) not null,
   DATA                 date,
   VALOR                numeric(5,0),
   TIPO_REP             varchar(15),
   ID_ORC               integer not null,
   primary key (ID_ORC)
)
type = InnoDB;

/*==============================================================
*/
/* Table: REPARACAO
*/
/*==============================================================
*/
create table REPARACAO
(
   ID_CORDAO            numeric(10,0),
   ID_REP               numeric(8,0),
   DATA_ENT             date,
   VALOR                numeric(5,0),
   PRONTO               bool,
   PAGO                 bool
)
type = InnoDB;

/*==============================================================
*/
/* Table: RESULTADOTESTE
*/
/*==============================================================
*/
create table RESULTADOTESTE
```

```
(
   ID_RT                int not null,
   ID_CORDAO            numeric(10,0),
   ID_ASA               int,
   DESVIO               float(3),
   primary key (ID_RT)
);

/*==============================================================
*/
/* Table: TESTE
*/
/*==============================================================
*/
create table TESTE
(
   ID_TESTE             numeric(10,0) not null,
   ID_UTIL              numeric(8,0),
   ID_CORDAO            numeric(10,0),
   DATA                 date,
   ALONG_5              float(10),
   ALONG_10             float(10),
   ALONG_15             float(10),
   ALONG_20             float(10),
   ALONG_25             float(10),
   ALONG_50             float(10),
   ALONG_75             float(10),
   ALONG_100            float(10),
   ALONG_125            float(10),
   ALONG_150            float(10),
   ALONG_175            float(10),
   ALONG_200            float(10),
   ALONG_225            float(10),
   DIAGNOSTICO          varchar(150),
   primary key (ID_TESTE)
)
type = InnoDB;

/*==============================================================
*/
/* Table: TOLERANCIACORDAO
*/
/*==============================================================
*/
create table TOLERANCIACORDAO
(
   ID_TC                int(8) not null,
   TIPOA                int(3),
   TIPOB                int(3),
   TIPOC                int(3),
   TIPOD                int(3)
);

/*==============================================================
*/
```

```
/* Table: TOLERANCIAMARCA
*/
/*==============================================================
*/
create table TOLERANCIAMARCA
(
   ID_TM                int(8) not null,
   MARCA                varchar(20),
   MB                   int(3),
   B                    int(3),
   S                    int(3),
   M                    int(3),
   primary key (ID_TM)
);

/*==============================================================
*/
/* Table: UTILIZADORES
*/
/*==============================================================
*/
create table UTILIZADORES
(
   ID_UTIL              numeric(8,0) not null,
   NOME                 varchar(50),
   TIPO                 varchar(20),
   USERNAME             varchar(15),
   PASSWORD             varchar(20),
   primary key (ID_UTIL)
)
type = InnoDB;

alter table CORDAO add constraint FK_POSSUI foreign key (ID_ORC)
      references ORCAMENTO (ID_ORC) on delete restrict on update
restrict;

alter table CORDAO add constraint FK_TEM foreign key
(ID_CLIENTE)
      references CLIENTES (ID_CLIENTE) on delete restrict on
update restrict;

alter table DADOSCORDOES add constraint FK_RELATIONSHIP_5
foreign key (ID_CORDAO)
      references CORDAO (ID_CORDAO) on delete restrict on update
restrict;

alter table DADOSCORDOES add constraint FK_RELATIONSHIP_6
foreign key (ID_DF)
      references DADOS_FABRICA (ID_DF) on delete restrict on
update restrict;

alter table ORCAMENTO add constraint FK_POSSUI2 foreign key
(ID_CORDAO)
      references CORDAO (ID_CORDAO) on delete restrict on update
restrict;
```

```
alter table REPARACAO add constraint FK_E foreign key
(ID_CORDAO)
      references CORDAO (ID_CORDAO) on delete restrict on update
restrict;

alter table RESULTADOTESTE add constraint FK_POSSUI3 foreign key
(ID_CORDAO)
      references CORDAO (ID_CORDAO) on delete restrict on update
restrict;

alter table TESTE add constraint FK_EFECTUA foreign key
(ID_CORDAO)
      references CORDAO (ID_CORDAO) on delete restrict on update
restrict;

alter table TESTE add constraint FK_FAZEM foreign key (ID_UTIL)
      references UTILIZADORES (ID_UTIL) on delete restrict on
update restrict;
```

Anexo D – Codificação da placa de aquisição – Arduíno

```
int pin = 0; // analog pin
int pin2 = 1;
double tempc = 0,tempf=0, tempd =0; // variáveis para os sensores
double samples[100]; // melhor precisão
int maxi = -100,mini = 100; //  max/min temperatura
int i;
int variavel;
char buffer[18];
char st1[10],st2[10];
char *virg =",";
String total;


void setup()
{
  Serial.begin(9600); // inicia comunicação
}

void loop()
{
  variavel =0;

  if (Serial.available() > 0)
  {
    variavel = Serial.read();
  }

  //Se recebeu o caracter 'a' significa que deverá enviar os dados
  //relativos aos sensores
  if(variavel=='a')
  {
  for(i = 0;i<=7;i++){ // gets 8 samples of temperature

    samples[i] = ( 5.0 * analogRead(pin) * 100.0) / 1024.0;
    tempc = tempc + samples[i];
  }
  tempc = tempc/8.0; // melhor precisão
  tempf = (tempc * 9)/ 5 + 32; // conversão para cº

  if(tempc > maxi) {
    maxi = tempc;
  } // set max temperature
  if(tempc < mini) {
    mini = tempc;
  } // set min temperature

  tempd = (5.0 * analogRead(pin2) * 100.0) / 10024.0;


//Como so podem ser enviados dados por um aporta serial
//são concatenados os dois valores do sensores
//e cabe à aplicação central posteriomente proceder à
//respetiva separação dado que sabe que vão separados por uma
//virgula
dtostrf(tempc, 0, 2, st1);
dtostrf(tempd, 0, 2, st2);
strcat (st1, virg);
strcat (st1,st2);

//Escreve no canal
Serial.println(st1);
tempc = 0;

  }
  variavel='b';
}
```

Anexo E – Excertos do Código da Aplicação Sistema de testes

Classe Comunicação

```
namespace DTO
{
    class Comunicacao
    {
        static SerialPort _serialPort;

        public SerialPort getserialPort()
        {
            return _serialPort;
        }
        public Comunicacao()
        {
            _serialPort = new SerialPort();
        }
        public void ConfLigacao(string PortName, int BR)
        {
            _serialPort.PortName = PortName;
            _serialPort.BaudRate = BR;
        }
        public bool AbrirPorta()
        {
            if (_serialPort.IsOpen == false)
                _serialPort.Open();
            if (_serialPort.IsOpen)
                return true;
            else
                return false;
        }

        public void FecharPorta()
        {
            _serialPort.Close();
        }

        public void Escreve(char[] buff)
        {
            _serialPort.Write(buff, 0, 1);
        }

        public string Le()
        {
            string receber;
            receber = _serialPort.ReadExisting();
            return receber;
        }
    }
}
```

Classe CordaoDAL

```
using System;
using System.IO;
using System.Collections;
using System.Globalization;
using MySql.Data.MySqlClient;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using ZedGraph;
using DTO;

namespace DAL
{
    public class CordaoDAL
    {
        private MySqlConnection con = null;
        String _conexaoMySQL = "";

        public CordaoDAL()
        {
            _conexaoMySQL = "server = localhost; user id=root;
database=db_mestrado";
        }

        //Mostar todos os cordões da tabela Cordão
        public DataTable selectCordoes()
        {
            try
            {
                String sql = "SELECT * FROM Cordao";
                con = new MySqlConnection(_conexaoMySQL);
                MySqlCommand cmd = new MySqlCommand(sql, con);
                MySqlDataAdapter da = new MySqlDataAdapter();
                da.SelectCommand = cmd;
                DataTable dt = new DataTable();
                da.Fill(dt);
                return dt;
            }
            catch (Exception ex)
            {
                throw ex;
            }
        }

        // Retorna uma unica entidade Cordao que tem o codigo = id
        public Cordao selectCordaoByID(int id)
        {
            try
            {
                String sql = "SELECT * FROM cordao WHERE id_cordao = @id";
                con = new MySqlConnection(_conexaoMySQL);
                MySqlCommand cmd = new MySqlCommand(sql, con);
                cmd.Parameters.AddWithValue("@id", id);
                con.Open();
                MySqlDataReader dr;
                Cordao cordao = new Cordao();
                dr = cmd.ExecuteReader(CommandBehavior.CloseConnection);
                while (dr.Read())
                {
                    cordao.setIdCordao(Convert.ToInt32(dr["id_cordao"]));
                    cordao.setIdCliente(Convert.ToInt32(dr["id_cliente"]));
                    cordao.setMarca(dr["marca"].ToString());
```

```
                    cordao.setTipo(dr["tipo"].ToString());
                    cordao.setRefe(dr["ref"].ToString());
                    cordao.setPosicao(dr["posicao"].ToString());
                    cordao.setEstado(dr["estado"].ToString());
                    cordao.setCores(dr["cores"].ToString());
                    cordao.setImagem(dr["imagem"].ToString());
                }
                return cordao;
            }
            catch (Exception ex)
            {
                throw ex;
            }
        }

        public void insertCordao(Cordao cordao)
        {
            try
            {
                string dfa = null;
                int proxidcordao = 0;

                String sql = "INSERT INTO cordao
(id_cliente,marca,tipo,ref,posicao,estado,cores,imagem)VALUES(@id_cliente,@mar
ca,@tipo,@ref,@posicao,@estado,@cores,@imagem)";
                con = new MySqlConnection(_conexaoMySQL);
                con.Open();
                MySqlCommand cmd = new MySqlCommand(sql, con);
                MySqlCommand cmdMySQL = con.CreateCommand();
                MySqlCommand cmdMySQL2 = con.CreateCommand();
                MySqlCommand cmdMySQL3 = con.CreateCommand();
                MySqlDataReader reader;
                cmdMySQL.Parameters.AddWithValue("@ref", cordao.getRefe());

                //Para saber qual o valor da tabela dadosfabrica corresponde
ao valor do cordao que estamos a inserir
                //Cada cordao que é inserido na base de dados tem um valor de
fábrica
                cmdMySQL.CommandText = "Select * from dadosfabrica where
reff=@ref ";
                //para saber qual o o ultimo id_cordao que foi atribuido para
desta formA sabermos qual o proximo que vai ser atribuido
                //dado que o campo id_cordao e autoincrement
                cmdMySQL3.CommandText = "select MAX(id_cordao) from cordao";
                reader = cmdMySQL.ExecuteReader();

                while (reader.Read())
                {
                    dfa = reader["id_df"].ToString();
                }
                reader.Close();
                reader = cmdMySQL3.ExecuteReader();
                while (reader.Read())
                {
                    proxidcordao =
(Convert.ToInt32(reader["MAX(id_cordao)"]));
                    MessageBox.Show("ai vaii e show" + proxidcordao);
                }
                reader.Close();
                proxidcordao = proxidcordao + 1;
                String sql2 = "Insert into dadoscordoes (id_cordao,id_df)
values ( " + proxidcordao + "," + dfa + " )";
                MessageBox.Show("String 2 " + sql2);
                MySqlCommand cm2d = new MySqlCommand(sql2, con);

                cmd.Parameters.AddWithValue("@id_cliente",
cordao.getId_Cliente());
                cmd.Parameters.AddWithValue("@marca", cordao.getMarca());
```

```
                cmd.Parameters.AddWithValue("@tipo", cordao.getTipo());
                cmd.Parameters.AddWithValue("@ref", cordao.getRefe());
                cmd.Parameters.AddWithValue("@posicao", cordao.getPos());
                cmd.Parameters.AddWithValue("@estado", cordao.getEstado());
                cmd.Parameters.AddWithValue("@cores", cordao.getCores());
                cmd.Parameters.AddWithValue("@imagem", cordao.getImagem());

                cmd.ExecuteNonQuery();
                cm2d.ExecuteNonQuery();
                MessageBox.Show("O seu registo foi inserido com sucesso");
            }
            catch (Exception ex)
            {
                MessageBox.Show("" + ex);
                throw ex;
            }
            finally
            {
                con.Close();
            }
        }

        // Preenche a combobox da form gerecordoes com os id_clientes's
existentes
        public DataTable PreencheComboCordoes()
        {
            string sql = "SELECT distinct id_cliente FROM cordao ORDER BY
ID_CLIENTE";
            con = new MySqlConnection(_conexaoMySQL);
            MySqlCommand cmdSel = new MySqlCommand(sql, con);
            DataTable dt = new DataTable();
            MySqlDataAdapter da = new MySqlDataAdapter(cmdSel);
            da.Fill(dt);
            return dt;
        }

        // Preenche a combobox da form geredadosfabrica com as ref´s
existentes na tabela dadosfabrica
        public DataTable PreencheComboRef()
        {
            string sql = "SELECT distinct reff FROM dadosfabrica ORDER BY
reff";
            con = new MySqlConnection(_conexaoMySQL);
            MySqlCommand cmdSel = new MySqlCommand(sql, con);
            DataTable dt = new DataTable();
            MySqlDataAdapter da = new MySqlDataAdapter(cmdSel);
            da.Fill(dt);
            return dt;
        }

        // Preenche a combobox da form geredadosfabrica com as ref´s
existentes na tabela dadosfabrica
        public DataTable PreencheComboIDCordoes(int idcli)
        {
            string sql = "SELECT * FROM cordao where id_cliente=" + idcli + "
ORDER BY id_cordao";
            //MessageBox.Show("sql " + sql);
            con = new MySqlConnection(_conexaoMySQL);
            MySqlCommand cmdSel = new MySqlCommand(sql, con);
            DataTable dt = new DataTable();
            MySqlDataAdapter da = new MySqlDataAdapter(cmdSel);
            da.Fill(dt);
            return dt;
        }

        // Preenche a combobox da form geredadosfabrica com as ref´s
existentes na tabela dadosfabrica
        public DataTable PreencheComboIDCliente()
```

```
        {
            string sql = "SELECT distinct id_cliente FROM cordao ORDER BY
id_cliente";
            con = new MySqlConnection(_conexaoMySQL);
            MySqlCommand cmdSel = new MySqlCommand(sql, con);
            DataTable dt = new DataTable();
            MySqlDataAdapter da = new MySqlDataAdapter(cmdSel);
            da.Fill(dt);
            return dt;
            //ComboBox1.DataSource = dt;
            //ComboBox1.DisplayMember = "Name";//column name to display
        }

        //altera cordao
        public void updateCordao(Cordao cordao)
        {
            try
            {
                String sql = "UPDATE cordao SET  marca=@marca, tipo=@tipo,
ref=@ref, posicao=@, estado=@estado, cores=@cores, imagem=@imagem  WHERE
id_cordao = @id_cordao ";
                con = new MySqlConnection(_conexaoMySQL);
                MySqlCommand cmd = new MySqlCommand(sql, con);
                cmd.Parameters.AddWithValue("@id_cordao",
cordao.getId_Cordao());
                cmd.Parameters.AddWithValue("@id_cliente",
cordao.getId_Cliente());
                cmd.Parameters.AddWithValue("@marca", cordao.getMarca());
                cmd.Parameters.AddWithValue("@tipo", cordao.getTipo());
                cmd.Parameters.AddWithValue("@ref", cordao.getRefe());
                cmd.Parameters.AddWithValue("@posicao", cordao.getPos());
                cmd.Parameters.AddWithValue("@estado", cordao.getEstado());
                cmd.Parameters.AddWithValue("@cores", cordao.getCores());
                cmd.Parameters.AddWithValue("@imagem", cordao.getImagem());
                con.Open();
                cmd.ExecuteNonQuery();
                MessageBox.Show("O seu registo foi alterado com sucesso!");
            }
            catch (Exception ex)
            {
                throw ex;
            }
            finally
            {
                con.Close();
            }
        }

        //Eliminar cordao
        public void eliminaCordao(Cordao cordao)
        {
            try
            {
                String sql = "DELETE FROM cordao WHERE id_cordao = @id ";
                con = new MySqlConnection(_conexaoMySQL);
                MySqlCommand cmd = new MySqlCommand(sql, con);
                cmd.Parameters.AddWithValue("@id", cordao.getId_Cordao());
                con.Open();
                cmd.ExecuteNonQuery();
                MessageBox.Show("O seu registo foi apagado com sucesso!");
            }
            catch (Exception ex)
            {
                throw ex;
            }
            finally
            {
                con.Close();
```

```
            }
        }

        // Carrega Array de cordoes com os valores que existem na Base de
dados
        public void ReadBd()
        {

            DataSet md = new DataSet();
            con = new MySqlConnection(_conexaoMySQL);
            con.Open();

            if (con.State == ConnectionState.Open)
            {

                MySqlCommand cmdMySQL = con.CreateCommand();
                //criar mySQL reeader object
                MySqlDataReader reader;
                //mySQL command object
                cmdMySQL.CommandText = "select * from cordao";
                //execute the reader, thus retrieving the data
                reader = cmdMySQL.ExecuteReader();

                // Como vai ler todos os registos da base de dados é
necessário limpar a lista
                Clientes.v_cord.Clear();
                while (reader.Read())
                {
                    Cordao c = new Cordao();
                    c.setIdCordao(Convert.ToInt16((reader["ID_CORDAO"])));
                    if (reader["id_cliente"].ToString() != "")

c.setIdCliente(Convert.ToInt16((reader["id_cliente"])));
                    if (reader["marca"].ToString() != "")
                        c.setMarca(reader["marca"].ToString());
                    if (reader["tipo"].ToString() != "")
                        c.setTipo(reader["tipo"].ToString());
                    if (reader["ref"].ToString() != "")
                        c.setRefe(reader["ref"].ToString());
                    if (reader["posicao"].ToString() != "")
                        c.setPosicao(reader["posicao"].ToString());
                    if (reader["estado"].ToString() != "")
                        c.setEstado(reader["estado"].ToString());
                    if (reader["cores"].ToString() != "")
                        c.setCores(reader["cores"].ToString());
                    if (reader["imagem"].ToString() != "")
                        c.setImagem(reader["imagem"].ToString());

                    Clientes.v_cord.Add(c);
                }
                //MessageBox.Show("Os dados forma carregado nas listas com
sucesso");
            }
            con.Close();
        }

        // insere na bd os dados que foram guardados nas lista durante a
execução do programa - CORDÃO
        public void InsertBD(string campo1, string campo2, string campo3,
string campo4, string campo5, string campo6, string campo7, string campo8,
string campo9)
        {
            string marca, tipo, reff, posicao, estado, cores, imagem;
            // abre a ligação
            con = new MySqlConnection(_conexaoMySQL);
            con.Open();
            // Se a ligação estiver aberta
            if (con.State == ConnectionState.Open)
```

```
            {
                try
                {
                    MySqlCommand commSS1 = new MySqlCommand(temp, con);
                    MySqlCommand commSS2 = new MySqlCommand(q, con);
                    MySqlCommand commSS3 = new MySqlCommand(temp2, con);
                    commSS1.ExecuteNonQuery();
                    commSS2.ExecuteNonQuery();
                    commSS3.ExecuteNonQuery();
                    //MessageBox.Show("Apaguei a BD");
                }
                catch (MySqlException ex)
                {
                    MessageBox.Show("Erro:!\n" + ex.Message,
                         "Informação", MessageBoxButtons.OK,
MessageBoxIcon.Error);
                }

                //Construir a string de inserção com a tabela e os campos
indicados
                string query = "INSERT INTO  cordao " + "(" + campo1 + "," +
campo2 + "," + campo3 + "," +
                campo4 + "," + campo5 + "," + campo6 + "," + campo7 + "," +
campo8 + "," + campo9 + ")" + " VALUES ";

                StringBuilder sb = new StringBuilder();
                //adicona o que ja consta na string query
                sb.Append(query);
                //ciclo for para percorrer a lista que contem os valores dos
sensores
                for (int i = 0; i < Clientes.v_cord.Count; i++)
                {
                    //Necessário verificar se o campo contém valor. Caso
contrário irá gerar erro ao efectuar a conversão
                    if (Clientes.v_cord[i].getMarca() != null)
                        marca = Clientes.v_cord[i].getMarca().ToString();
                    else
                        marca = null;
                    if (Clientes.v_cord[i].getTipo() != null)
                        tipo = Clientes.v_cord[i].getTipo().ToString();
                    else
                        tipo = null;
                    if (Clientes.v_cord[i].getRefe() != null)
                        reff = Clientes.v_cord[i].getRefe().ToString();
                    else
                        reff = null;
                    if (Clientes.v_cord[i].getPos() != null)
                        posicao = Clientes.v_cord[i].getPos().ToString();
                    else
                        posicao = null;
                    if (Clientes.v_cord[i].getEstado() != null)
                        estado = Clientes.v_cord[i].getEstado().ToString();
                    else
                        estado = null;
                    if (Clientes.v_cord[i].getCores() != null)
                        cores = Clientes.v_cord[i].getCores().ToString();
                    else
                        cores = null;
                    if (Clientes.v_cord[i].getImagem() != null)
                        imagem = Clientes.v_cord[i].getImagem().ToString();
                    else
                        imagem = null;

                    sb.Append("(" + Clientes.v_cord[i].getId_Cordao() + "," +
Clientes.v_cord[i].getId_Cliente() + "," + "'" + marca + "'" + "," + "'" +
tipo + "'" + "," + "'" + reff + "'" + "," + "'" + posicao + "'" + "," + "'" +
estado + "'" + "," + "'" + cores + "'" + "," + "'" + imagem + "'" + ")");
                    if (i != Clientes.v_cord.Count() - 1)
```

```
                    sb.Append(",");

                }

                //  MessageBox.Show(sb.Length.ToString());
                // acrescenta se um ; para finalizar a query
                sb.Append(";");
                String queryF = sb.ToString();
                // Executa-se a query que foi construida para que os dados
sejam introduzidos na base de dados
                MySqlCommand commS = new MySqlCommand(queryF, con);
                try
                {
                    commS.ExecuteNonQuery();
                }
                catch (MySqlException ex)
                {
                    MessageBox.Show("Erro:!\n" + ex.Message,
                        "Informação", MessageBoxButtons.OK,
MessageBoxIcon.Error);
                }
                // Fecha a conexão
                con.Close();

            }
        }
    }
}
```

Código relativo à execução de um teste

```
   private void Form2_Load(object sender, EventArgs e)
        {
            // Preencher a combox que esta na parte das configuracoes com as
ref´s que existem na tabela dados de fabrica
            comboBox4.Enabled = false;
            BLL.CordaoBLL cbll = new CordaoBLL();
            dt = cbll.PreencheComboRef();
            BLL.ClienteBLL clibll = new ClienteBLL();

            dt3 = cbll.PreencheComboIDCliente();
            dt4 = clibll.PreencheComboNOME();
            comboBox2.DataSource = dt4;
            comboBox2.DisplayMember = "nome";
            comboBox2.ValueMember = "nome";

            GraphPane myPane = zedGraphControl1.GraphPane;
            myPane.Title.Text = "Gráfico relativo à Pressão\n";
            myPane.XAxis.Title.Text = "Tempo, Segundos";
            myPane.YAxis.Title.Text = "Pressão, daN";

            // 2 graph
            GraphPane myPane2 = zedGraphControl2.GraphPane;
            myPane2.Title.Text = "Gráfico relativo ao Alongamento\n";
            myPane2.XAxis.Title.Text = "Tempo, Segundos";
            myPane2.YAxis.Title.Text = "Alongamento, %";

            // 3 graph
            GraphPane myPane3 = zedGraphControl3.GraphPane;
            myPane3.Title.Text = "Alongamento/Pressão";
            myPane3.XAxis.Title.Text = "Alongamento, %";
            myPane3.YAxis.Title.Text = "Pressão, daN";

            // The RollingPointPairList is an efficient storage class that
always
            // keeps a rolling set of point data without needing to shift any
data values
            RollingPointPairList list = new RollingPointPairList(1200);
            RollingPointPairList list2 = new RollingPointPairList(1200);
            RollingPointPairList list3 = new RollingPointPairList(1200);
            //RollingPointPairList list4 = new RollingPointPairList(1200);
            //RollingPointPairList list5 = new RollingPointPairList(1200);

            // Color is blue, and there will be no symbols
            LineItem curve = myPane.AddCurve("Alongamento", list, Color.Blue,
SymbolType.None);
            //2 graph
            curve = myPane2.AddCurve("Pressão", list2, Color.Red,
SymbolType.None);
            //3 graph
            curve = myPane3.AddCurve("Pressão /alongamento", list3, Color.Red,
SymbolType.None);

            // Just manually control the X axis range so it scrolls
continuously
            // instead of discrete step-sized jumps
            myPane.XAxis.Scale.Min = 0;
            myPane.XAxis.Scale.Max = 30;
            myPane.XAxis.Scale.MinorStep = 1;
            myPane.XAxis.Scale.MajorStep = 5;

            //2 graph
            // Just manually control the X axis range so it scrolls
continuously
            // instead of discrete step-sized jumps
            myPane2.XAxis.Scale.Min = 0;
            myPane2.XAxis.Scale.Max = 30;
```

```
            myPane2.XAxis.Scale.MinorStep = 1;
            myPane2.XAxis.Scale.MajorStep = 5;
            //3 graph
            // Just manually control the X axis range so it scrolls
continuously
            // instead of discrete step-sized jumps
            myPane3.XAxis.Scale.Min = 0;
            myPane3.XAxis.Scale.Max = 30;
            myPane3.XAxis.Scale.MinorStep = 1;
            myPane3.XAxis.Scale.MajorStep = 5;

            // Scale the axes
            zedGraphControl1.AxisChange();
            //2 graph
            // Scale the axes
            zedGraphControl2.AxisChange();
            //3 graph
            // Scale the axes
            zedGraphControl3.AxisChange();
        }

        // Set the size and location of the ZedGraphControl
        private void SetSize()
        {
            // Control is always 10 pixels inset from the client rectangle of
the form
            Rectangle formRect = this.ClientRectangle;
            formRect.Inflate(-100, -100);
            if (zedGraphControl1.Size != formRect.Size)
            {
                zedGraphControl1.Location = formRect.Location;
                zedGraphControl1.Size = formRect.Size;
            }
            //2 graph
            if (zedGraphControl2.Size != formRect.Size)
            {
                zedGraphControl2.Location = formRect.Location;
                zedGraphControl2.Size = formRect.Size;
            }
            //3 graph
            if (zedGraphControl3.Size != formRect.Size)
            {
                zedGraphControl3.Location = formRect.Location;
                zedGraphControl3.Size = formRect.Size;
            }
        }


        private void timer1_Tick(object sender, EventArgs e)
        {
            // Make sure that the curvelist has at least one curve
            if (zedGraphControl1.GraphPane.CurveList.Count <= 0)
                return;

            // Get the first CurveItem in the graph
            LineItem curve = zedGraphControl1.GraphPane.CurveList[0] as
LineItem;
            if (curve == null)
                return;

            // Make sure that the curvelist has at least one curve
            if (zedGraphControl2.GraphPane.CurveList.Count <= 0)
                return;

            // Get the first CurveItem in the graph
            LineItem curve2 = zedGraphControl2.GraphPane.CurveList[0] as
LineItem;
            if (curve2 == null)
```

```
                return;

            // Make sure that the curvelist has at least one curve
            if (zedGraphControl3.GraphPane.CurveList.Count <= 0)
                return;

            // Get the first CurveItem in the graph
            LineItem curve3 = zedGraphControl3.GraphPane.CurveList[0] as
LineItem;
            if (curve3 == null)
                return;

            // Get the PointPairList
            curve.Clear();
            curve2.Clear();
            curve3.Clear();
            //curve4.Clear();
            IPointListEdit list = curve.Points as IPointListEdit;
            IPointListEdit list2 = curve2.Points as IPointListEdit;
            IPointListEdit list3 = curve3.Points as IPointListEdit;

            // If this is null, it means the reference at curve.Points does
not
            // support IPointListEdit, so we won't be able to modify it
            if (list == null)
                return;
            if (list2 == null)
                return;
            if (list3 == null)
                return;

            int tam = arrText_ALONG.Count;
            int tamP = arrText_PRESSAO.Count;
            // Random rnd = new Random();
            for (int i = 1; i < tam; i++)
            {

                list.Add(Convert.ToDouble(arrText_TEMPOG1[i]),
Convert.ToDouble(arrText_ALONG[i]));
                list2.Add(Convert.ToDouble(arrText_TEMPOG2[i]),
Convert.ToDouble(arrText_PRESSAO[i]));
                list3.Add(Convert.ToDouble(arrText_PRESSAO[i]),
Convert.ToDouble(arrText_ALONG[i]));


            }

            // Keep the X scale at a rolling 30 second interval, with one
            // major step between the max X value and the end of the axis
            Scale xScale = zedGraphControl1.GraphPane.XAxis.Scale;
            double time = (Environment.TickCount - tickStart) / 1000.0;
            if (time > xScale.Max - xScale.MajorStep)
            {
                xScale.Max = time + xScale.MajorStep;
                xScale.Min = xScale.Max - 30.0;
            }
            //2 graph
            Scale xScale2 = zedGraphControl2.GraphPane.XAxis.Scale;

            if (time > xScale2.Max - xScale2.MajorStep)
            {
                xScale2.Max = time + xScale2.MajorStep;
                xScale2.Min = xScale2.Max - 30.0;
            }

            //3 graph
            Scale xScale3 = zedGraphControl3.GraphPane.XAxis.Scale;
```

```
            if (time > xScale3.Max - xScale3.MajorStep)
            {
                xScale3.Max = time + xScale3.MajorStep;
                xScale3.Min = xScale3.Max - 30.0;
            }

            Scale yScale = zedGraphControl1.GraphPane.YAxis.Scale;
            yScale.Max = MaximoY + 0.5;
            yScale.Min = MinimoY - 0.5;
            // Make sure the Y axis is rescaled to accommodate actual data
            zedGraphControl1.AxisChange();
            // Force a redraw
            zedGraphControl1.Invalidate();

            //2 graph
            Scale yScale2 = zedGraphControl2.GraphPane.YAxis.Scale;
            yScale2.Max = MaximoY2 + 0.5;
            yScale2.Min = MinimoY2 - 0.5;
            // Make sure the Y axis is rescaled to accommodate actual data
            zedGraphControl2.AxisChange();
            // Force a redraw
            zedGraphControl2.Invalidate();

            //3 graph
            Scale yScale3 = zedGraphControl3.GraphPane.YAxis.Scale;
            yScale3.Max = MaximoY3 + 0.5;
            yScale3.Min = MinimoY3 - 0.5;
            // Make sure the Y axis is rescaled to accommodate actual data
            zedGraphControl3.AxisChange();
            // Force a redraw
            zedGraphControl3.Invalidate();
        }

        private void Form2_Resize(object sender, EventArgs e)
        {
            SetSize();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            timer1.Enabled = true;
        }

        private void button2_Click(object sender, EventArgs e)
        {
            timer1.Enabled = false;
            grafest();


        }

        ////////////////////////////////////////////////////////////////////
        //PARTE DOS SENSORES

        private void startSensor()
        {
            DAL.TesteDAL tdal = new TesteDAL();
            string receber;
            //int i=0;
            double P = 0;
            Inicia_Sensor();
            while (flag == 1)
            {
                if (sens.getserialPort().IsOpen)
                {
                    // Se a porta esta aberta declara uma char[] com um
elemento.
                    char[] buff = new char[1];
```

```
                    // Caarega a variavel com o elemento.
                    buff[0] = 'a';
                    // Envia o buffer com uma variavel
                    sens.Escreve(buff);
                    //espera um segundo para receber novos dados
                    PauseForMilliSeconds(1000);
                    //Le do canal serial o valor
                    receber = sens.Le();
                    if (receber != "")
                    {
                        //Tem de separar os valores dos sensores uma vez que
vêm numa só variável
                        string[] words = receber.Split(',');
                        string[] words1 = receber.Split(',');
                        string[] words2 = words[1].Split('/');
                        words[0] = words[0].Replace(".", ",");
                        words2[0] = words2[0].Replace(".", ",");

                        //  MessageBox.Show("temp = " + receber);
                        if (receber == "")
                        {
                            //MessageBox.Show("Pressao = " + receber);
                        }
                        else
                        {
                            double temp = Convert.ToDouble(words[0]);
                            int guardaPressao = Convert.ToInt32(temp);
                            //double P = 12.01;
                            P += 1;
                            //Convert.ToDouble(words2[0]);
                            if (guardaPressao == 5 || guardaPressao == 10 ||
guardaPressao == 15 || guardaPressao == 20 || guardaPressao == 25 ||
guardaPressao == 50 || guardaPressao == 75 || guardaPressao == 100 ||
guardaPressao == 125 || guardaPressao == 150 || guardaPressao == 175 ||
guardaPressao == 200 || guardaPressao == 225)
                                // chamafuncão de testes pa guardar o
valor!!!!
                                tdal.selectDados(guardaPressao, temp);
                            arrText_ALONG.Add(temp);
                            arrText_PRESSAO.Add(P);

                            double time = (Environment.TickCount - tickStart)
/ 1000.0;
                            textBox1.Text = temp.ToString();
                            textBox4.Text = P.ToString();
                            textBox3.Text = time.ToString();
                            textBox5.Text = time.ToString();
                            arrText_TEMPOG1.Add(time);
                            arrText_TEMPOG2.Add(time);
                            //Para escalar os gráficos automaticamente em
função dos valores recebidos
                            if (primeiro)
                            {
                                MaximoY = MinimoY = temp;
                                primeiro = false;
                            }
                            else
                            {
                                if (MaximoY < temp)
                                    MaximoY = temp;
                                if (MinimoY > temp)
                                    MinimoY = temp;
                            }

                            if (primeiro2)
                            {
                                MaximoY2 = MinimoY2 = P + 50;
                                primeiro2 = false;
```

```
                            }
                            else
                            {
                                if (MaximoY2 < P)
                                    MaximoY2 = P + 50;
                                if (MinimoY2 > P)
                                    MinimoY2 = P;
                            }
                            if (primeiro3)
                            {
                                MaximoY3 = MinimoY3 = temp;
                                primeiro3 = false;
                            }
                            else
                            {
                                if (MaximoY3 < temp)
                                    MaximoY3 = temp;
                                if (MinimoY3 > temp)
                                    MinimoY3 = temp;
                            }

                            if (primeiro3)
                            {
                                MaximoY3 = MinimoY3 = temp;
                                primeiro3 = false;
                            }
                            else
                            {
                                if (MaximoY3 < temp)
                                    MaximoY3 = temp;
                                if (MinimoY3 > temp)
                                    MinimoY3 = temp;
                            }
                        }
                    }
                    else
                    {
                        //MessageBox.Show("Porta nao esta aberta");
                    }
                }
            }
        }

        public static DateTime PauseForMilliSeconds(int
MilliSecondsToPauseFor)
        {
            System.DateTime ThisMoment = System.DateTime.Now;
            System.TimeSpan duration = new System.TimeSpan(0, 0, 0, 0,
MilliSecondsToPauseFor);
            System.DateTime AfterWards = ThisMoment.Add(duration);

            while (AfterWards >= ThisMoment)
            {
                System.Windows.Forms.Application.DoEvents();
                ThisMoment = System.DateTime.Now;
            }
            return System.DateTime.Now;
        }

        private void button4_Click(object sender, EventArgs e)
        {
            flag = 1;
            tickStart = Environment.TickCount;
            startSensor();
        }
        private void Inicia_Sensor()
        {
            sens.ConfLigacao("COM8", 9600);
```

```
            sens.AbrirPorta();
        }

        private void Para_Sensor()
        {
            sens.FecharPorta();
        }



        private void timer2_Tick(object sender, EventArgs e)
        {
            startSensor();
        }

        private void button3_Click(object sender, EventArgs e)
        {
            flag = 0;
            serialPort1.Close();
        }

        public void grafest()
        {
            Cordao cord = new Cordao();
            cord.getVectorDFabrica();

            //Precisa verificar quais sao os dados de fabrica do cordao em
teste
            //através da referencia do cordao
            string r = (String)selectedDataRow3["ref"];
            //MessageBox.Show("a ref e " + r);
            BLL.DadosFabricaBLL dfbll = new DadosFabricaBLL();
            arrTesteDF = dfbll.procuraRef(r);
            //MessageBox.Show("dados da fabrica" + arrTesteDF[0].ToString());

            arrTesteAUX.Add(5);
            arrTesteAUX.Add(10);
            arrTesteAUX.Add(15);
            arrTesteAUX.Add(20);
            arrTesteAUX.Add(25);
            arrTesteAUX.Add(50);
            arrTesteAUX.Add(75);
            arrTesteAUX.Add(100);
            arrTesteAUX.Add(125);
            arrTesteAUX.Add(150);
            arrTesteAUX.Add(175);
            arrTesteAUX.Add(200);
            arrTesteAUX.Add(225);

            // Get a reference to the GraphPane instance in the
ZedGraphControl
            GraphPane myPane4 = zedGraphControl4.GraphPane;
            myPane4.Title.Text = "Alongamento/Pressão";
            myPane4.XAxis.Title.Text = "Pressão,daN";
            myPane4.Y2Axis.Title.Text = "Alongamento, Teste";
            myPane4.YAxis.Title.Text = "Alongamento, Dados Fábrica";

            // Make up some data points based on the Sine function
            PointPairList list4 = new PointPairList();
            PointPairList list5 = new PointPairList();

            // MessageBox.Show("tamanho do vector da BD com dados de fabrica"
+ arrTesteDF.Count);
            //MessageBox.Show("tamanho do vetor com dados do testes" +
DAL.TesteDAL.temp.Count);

            //o ciclo dura ate ao numero de valores que existem no dados de
fabrica
```

```
            //Alguns cordões apresentam valores para níveis de pressão que
outros nãp apresentam
            //exemplo:um cordão mais fino não é submetido a cargas tão
elevadas como um cordão mais grosso
            for (int i = 0; i < arrTesteDF.Count; i++)
            {
                list4.Add(Convert.ToDouble(arrTesteAUX[i]),
Convert.ToDouble(arrTesteDF[i]));
                list5.Add(Convert.ToDouble(arrTesteAUX[i]),
DAL.TesteDAL.temp[i]);
            }

            // Generate a red curve with diamond symbols, and "Alpha" in the
legend
            LineItem myCurve = myPane4.AddCurve("Dados Fábrica",
                list4, Color.Red, SymbolType.Diamond);
            // Fill the symbols with white
            myCurve.Symbol.Fill = new Fill(Color.White);

            // Generate a blue curve with circle symbols, and "Beta" in the
legend
            myCurve = myPane4.AddCurve("Teste",
                list5, Color.Blue, SymbolType.Circle);
            // Fill the symbols with white
            myCurve.Symbol.Fill = new Fill(Color.White);
            // Associate this curve with the Y2 axis
            myCurve.IsY2Axis = true;

            // Show the x axis grid
            myPane4.XAxis.MajorGrid.IsVisible = true;

            // Make the Y axis scale red
            myPane4.YAxis.Scale.FontSpec.FontColor = Color.Red;
            myPane4.YAxis.Title.FontSpec.FontColor = Color.Red;
            // turn off the opposite tics so the Y tics don't show up on the
Y2 axis
            myPane4.YAxis.MajorTic.IsOpposite = false;
            myPane4.YAxis.MinorTic.IsOpposite = false;
            // Don't display the Y zero line
            myPane4.YAxis.MajorGrid.IsZeroLine = false;
            // Align the Y axis labels so they are flush to the axis
            myPane4.YAxis.Scale.Align = AlignP.Inside;
            // Manually set the axis range
            myPane4.YAxis.Scale.Min = 0;
            myPane4.YAxis.Scale.Max = 30;


            // Enable the Y2 axis display
            myPane4.Y2Axis.IsVisible = true;
            // Make the Y2 axis scale blue
            myPane4.Y2Axis.Scale.FontSpec.FontColor = Color.Blue;
            myPane4.Y2Axis.Title.FontSpec.FontColor = Color.Blue;
            // turn off the opposite tics so the Y2 tics don't show up on the
Y axis
            myPane4.Y2Axis.MajorTic.IsOpposite = false;
            myPane4.Y2Axis.MinorTic.IsOpposite = false;
            // Display the Y2 axis grid lines
            myPane4.Y2Axis.MajorGrid.IsVisible = true;
            // Align the Y2 axis labels so they are flush to the axis
            myPane4.Y2Axis.Scale.Align = AlignP.Inside;
            myPane4.Y2Axis.Scale.Max = 30;
            myPane4.Y2Axis.Scale.Min = 0;

            // Fill the axis background with a gradient
            myPane4.Chart.Fill = new Fill(Color.White, Color.LightGray,
45.0f);

            // Tell ZedGraph to calculate the axis ranges
```

```
            // Note that you MUST call this after enabling IsAutoScrollRange,
since AxisChange() sets
            // up the proper scrolling parameters
            zedGraphControl4.AxisChange();
            // Make sure the Graph gets redrawn
            zedGraphControl4.Invalidate();
        }

        private void button5_Click(object sender, EventArgs e)
        {
            Relatorio r = new Relatorio();
            r.Show();
        }

        private void Form2_FormClosing(object sender, FormClosingEventArgs e)
        {
            DAL.TesteDAL tdal = new TesteDAL();

            int auxidcordao = Convert.ToInt32(selectedDataRow3["id_cordao"]);
            //Guarda os teste efetuado ao cordao na base de dados
            tdal.guardaTeste(auxidcordao);
        }

        private void comboBox4_SelectionChangeCommitted(object sender,
EventArgs e)
        {
            selectedDataRow3 = ((DataRowView)comboBox4.SelectedItem).Row;
        }

        private void button6_Click(object sender, EventArgs e)
        {
            DAL.TesteDAL tdal = new TesteDAL();
            tdal.mostraarray();
        }

        private void comboBox2_SelectionChangeCommitted_1(object sender,
EventArgs e)
        {
            comboBox4.Enabled = true;
            BLL.ClienteBLL clientebll = new ClienteBLL();
            DataRow selectedDataRow2 =
((DataRowView)comboBox2.SelectedItem).Row;
            string aux = selectedDataRow2["nome"].ToString();
            dt2 = clientebll.PreencheComboIDCordoes(aux);
            comboBox4.DataSource = dt2;
            comboBox4.DisplayMember = "id_cordao";
            comboBox4.ValueMember = "id_cordao";
        }

        private void button7_Click(object sender, EventArgs e)
        {
            timer1.Enabled = true;
            flag = 1;
            tickStart = Environment.TickCount;
            startSensor();
        }

        private void button8_Click(object sender, EventArgs e)
        {
            timer1.Enabled = false;
            grafest();
            flag = 0;
            serialPort1.Close();
        }

    }
}
```