Faculdade de Engenharia da Universidade do Porto

FEUP

Sistema de Supervisão e Controlo de Estruturas - Civiónica

Pedro Manuel Barbosa Moreira

VERSÃO PROVISÓRIA

Dissertação realizada no âmbito do
Mestrado Integrado em Engenharia Electrotécnica e de Computadores
Major Automação

Orientador: Prof. Dr. Adriano da Silva Carvalho

Junho de 2011

Aos meus Pais e Irmã

Resumo

Civiónica, um termo referente à combinação entre a Engenharia de Electrónica e a Engenharia Civil, é uma área interdisciplinar, que emerge da necessidade da utilização de sistemas de electrónica, em sistemas de monitorização de estruturas de engenharia civil. Com efeito, alude sobre a percepção da sociedade para as consequências sociais e económicas, derivadas da ocorrência de avarias ou acidentes graves em grandes obras estruturais. Facto que obriga a uma maior consciencialização dos responsáveis pelas obras, relativamente à importância desta monitorização estrutural. Tendo-se por isso, verificado nos últimos anos, um desenvolvimento tecnológico notável, ao nível dos sistemas de monitorização estrutural.

Com esta dissertação, pretende-se contribuir para esse desenvolvimento, sendo estudadas e testadas tecnologias existentes, que ainda não foram aplicadas nestes sistemas, mas que serão passíveis de utilização futura nos mesmos. E prova disso é que, desenvolveu-se um protótipo de um sistema de supervisão e controlo de estruturas, utilizando essas tecnologias.

Desta forma, realizou-se numa primeira abordagem, um estudo relativo a sistemas de supervisão e controlo em geral e à sua aplicação em estruturas de engenharia civil. Seguido, de um estudo teórico dos sistemas SCADA, focando os elementos que o constituem. Estudos estes, que conduziram a um levantamento e posterior análise, dos requisitos impostos a estes sistemas de monitorização, com os quais se depreendeu que uma arquitectura de processamento distribuído é a melhor solução para um sistema de monitorização de estruturas de grandes dimensões.

Face aos requisitos acima mencionados, surgiu também a necessidade de estudar as tecnologias de comunicação necessárias para a interligação entre todos os elementos do sistema, sendo inferido que uma boa solução para estas redes de comunicação são as redes CAN-Bus e as redes sem fios (*wireless*).

Além disso, explicou-se a implementação de todos os componentes do protótipo do sistema de supervisão e controlo de estruturas desenvolvido, no âmbito deste projecto: interface homem – máquina (HMI) fixa e Web; rede CAN-Bus; unidades de processamento e controlo; rede *wireless* 802.15.4; e unidades de processo/campo *wireless*.

Foi ainda, efectuada uma análise dos sensores e circuitos de condicionamento de sinal, utilizados nas unidades de processo do protótipo do sistema, os quais são: extensómetros; acelerómetros e sensores de temperatura. Sendo que, nestas unidades também foram utilizados actuadores representados com leds.

Por fim, apresentam-se e discutem-se os resultados relativos à análise das tecnologias de comunicações utilizadas, assim como, resultados obtidos em ensaios dos sensores.

Abstract

Civionics is a term which refers to the combination between electronic and civil engineering. It is an interdisciplinary field that emerges from the need to use electronics in structural health monitoring (SHM) of civil engineering. As a result, it draws society's attention to the social and economical consequences, derived from the occurrence of malfunctions or serious accidents within big structural constructions. This actually demands a stronger awareness towards the importance of this SHM. So, in the past few years, we have been attending a notable technological development in what concerns SHM.

With this dissertation, it is intended to contribute to that development, by studying and testing existent technologies that have not yet been applied to these systems, but that are likely to be used in the future. The proof of that is that, it has been developed a prototype of supervision and control systems of structures, using these very same technologies.

Thus, a first approach has been made with a study, which is related to supervision and control systems in general and its application in structures of civil engineering. This was followed by a theoretical study of the SCADA systems, focusing on the elements that are part of it. These studies led to the gathering and ulterior analysis of the imposed requirements for these monitoring systems, with which it is perceived that an architecture of distributed process, is the best possible solution for a monitoring system of big structural dimensions.

From the requirements mentioned above, there has also emerged a necessity to study the communication technologies, that are needed for the connection among all the elements of the system and it has been acknowledged that a good solution for these communication networks are the CAN-Bus networks, as well as, the wireless networks.

Besides, it is explained the implementation of all the components from the supervision and control systems of engineering structures prototype that has been developed in this project: human machine interface (HMI) fixed and Web; CAN-Bus network; processing and control units; wireless network 802.15.4 and process/field wireless units.

In this dissertation it has also been made the analysis of the sensors and signal conditioning circuits that are used in the process units of the system prototype, which are extensometers, accelerometers and temperature sensors. In these units, actuators were also used, represented by leds.

Finally, it is also presented and discussed some the results that are related to the communication technologies, such as, the results that were obtained testing the sensors.

Agradecimentos

Gostaria de deixar o meu profundo agradecimento a todos os que me auxiliaram na realização desta dissertação, pela disponibilidade, dedicação, paciência e compreensão, fundamentais para os resultados que aqui apresento. Ainda um reconhecimento especial a todos aqueles que me acompanharam ao longo destes últimos anos de formação académica.

Ao Professor Doutor Adriano Carvalho, orientador da dissertação, pelos conhecimentos transmitidos e por todo o apoio que sempre prestou durante a execução do projecto.

Ao colega e amigo João Ferreira, por me ter abraçado no mundo da Civiónica e por toda a ajuda e espírito de trabalho em equipa, fundamentais para a conclusão do projecto com sucesso.

Ao colega e amigo José Luís Pereira, que pela partilha dos seus vastos conhecimentos em Informática, muitas vezes me auxiliou na resolução de problemas de forma rápida e eficaz.

A todos os meus colegas efectivos e frequentadores do I105, pela partilha constante de conhecimentos e de emoções, que transformaram quase num Lar o nosso local de trabalho.

A todos os meus amigos, que de alguma forma contribuíram para que estes últimos cinco anos, tenham sido marcantes e inesquecíveis. Sérgio Luís, Emanuel Damaso, Marcelo Ferreira, António Monteiro, Isabel Silva, Pedro Pinto, Tânia Azevedo, Joana Azevedo, Luís Couto, Pedro Pinheiro, Paula Barreto, Ana Ferreira, Gabriel Damaso, Nuno Medon, Sara Tomás, Ana Carolina Silva, Carla Costa, Ana Catarina Silva, Tiago Silva, Cristiana Madureira, Tiago da Rocha, Tiago Ramos, Rui Barbosa, Henrique Teixeira, José Xavier, Alexandre Santos, Nuno Ferreira, Juliana Coutinho, Manuel Santos, Ângelo Vieira, João Sousa, Neuza Gomes, Sara Silva, André Moreira e ao Soares. A todos vocês, um muito obrigado!

À Tânia Oliveira, por toda a compreensão, preocupação, confiança, força, companhia, alegria, amizade e carinho. Obrigado por teres feito, de uma forma tão especial, parte deste ano de tantas mudanças.

Por fim aos meus Pais e Irmã, a quem dedico este trabalho. Pela compreensão, suporte, estímulo académico, apoio e carinho incondicional nesta e em tantas outras etapas de minha vida.

Índice

Lista de Figuras

Lista de Tabelas

Abreviaturas e Símbolos

Lista de abreviaturas

ACK	*Acknowledge*
ADC	*Analog-to-Digital Converter*
AP	Access Point
API	*Application Programming Interface*
ASCII	*American Standard Code for Information Interchange*
BSS	Basic Service Set
CA	Collision Avoidance
CAN	Controller Area Network
CAN	*Controller Area Network*
CANH	*CAN High*
CANL	*CAN Low*
CiA	CAN in Automation
CRC	*Cyclic Redundancy Check*
CSMA	Carrier Sense Multiple Access
DAN	*Device Area Network*
DCF	Distributed Coordination Function
DLC	*Data Length Code*
DSSS	*DSSS Direct Sequence Spread Spectrum*
E/S	*Entradas / Saídas*
ECC	*Execution Control Chart*
EMG	*Emergency Object*
EOF	*End of Frame*
ESS	*Extended Service Set*
FB	*Function Block*
FHSS	*Frequency Hopping Spread Spectrum*
FTU	*Fault Tolerance Unit*
GWT	*Google Web Toolkit*

HMI	H*uman-Machine Interaction*
IC	Integrated Circuit
IDE	*Identifier Extension*
IEC	*International Electrotechnical Commission*
IP	*Internet Protocol*
IR	Infrared
ISM	*Industrial, Scientific and Medical*
ISO	*International Organization for Standardization*
L2CAP	*Logical Link Controlol and Adaptation Protocol*
LAN	*Local Area Network*
MEMS	*Microelectromechanical Systems*
MOb	*Message Object*
MTU	*Master Terminal* Unit
NASA	*National Aeronautic and Space Administration*
NMT	*Network Management*
NTC	*Negative Temperature Coefficient*
OFDM	*Orthogonal Frequency Division Multiplexing*
OPC	*Object Linking and Embedding for Process Control*
OSI	*Open Systems Interconnection*
PAN	Personal Area Network
PCF	Point Coordination Function
PDO	*Process Data Object*
PLC	*Programmable Logic Controller*
PRT	*Platinum Resistance Thermometer*
PTC	*Positive Temperature Coefficient*
REC	*Receive Counter Error*
RF	Radio Frequency
RTC	*Real Time Clock*
RTD	*Resistance Temperature Detector*
RTR	*Remote Transmission Request*
RTU	*Remote Terminal Unit*
SAE	*Society Automotive Engineers*
SCADA	*Supervisory Control and Data Acquisition*
SD	*Secure Digital*
SDO	*Service Data Object*
SDP	*Service Discorevy Protocol*
SHM	*Structural Health Monitoring*
SI	*Système International d'Unités*
SOF	*Start of Frame*
SRR	*Substitute Remote Request*

SYNC	*Synchronization Object*
TCP	*Transmission Control Protocol*
TEC	*Transmit Counter Error*
TQ	*Time Quantum*
UART	*Universal Asynchronous Receiver/Transmitter*
UDP	User Datagram Protocol
WAN	*Wire Area Network*
WLAN	*Wireless Local Area Network*
ZC	*ZigBee Coordinator*
ZED	*ZigBee End Device*
ZR	*ZigBee Router*

Lista de símbolos

Ω	*Ohm* – unidade de medida de resistências eléctricas
ε	Extensão

Capítulo 1

Introdução

1.1 - Âmbito da Dissertação

Sistemas de supervisão e controlo são nos dias de hoje ferramentas quase obrigatórias em qualquer sistema de automação. Estes, permitem o acesso, a gestão e a obtenção de dados do processo, bem como, a possibilidade de o controlar e actuar directamente sobre ele. A sua gama de aplicações vai desde aplicação no âmbito industrial, quer para industriais de controlo discreto ou contínuo; aplicação em equipamentos de serviços públicos, como estações de tratamento de águas residuais, sistemas de transmissão de energia, sistemas de transportes; passando também por aplicações na área da domótica; e mais recentemente são utilizados também em estruturas de engenharia civil, como pontes ou edifícios.

Por conseguinte, um sistema de monitorização e controlo estrutural representa uma ferramenta muito poderosa, que pode permitir a obtenção de estruturas sustentáveis. Com estas tecnologias, tragédias como a da ponte Hintze Ribeiro de Entre-os-Rios, que caiu a 4 de Março de 2001 devido à cedência de um pilar, provocando a morte de 59 pessoas, poderão ser previstas, ao ponto de serem evitadas.

Parte constituinte dos sistemas de supervisão e controlo são os sistemas SCADA (*Supervisory Control and Data Acquisition*). Estes permitem a recolha de dados de vários sensores espalhados pelos vários constituintes do processo, centralizando-os num único computador que os processa e os apresenta ao operador. Permitem ainda: a detecção rápida de funcionamentos anómalos do processo em questão, para que possam ser corrigidos o mais rapidamente possível; e a actuação no processo por comandos enviados pelo operador, ou autonomamente pelo sistema configurado. Para além de que através da utilização de uma base de dados, estes sistemas permitem guardar um histórico das variáveis obtidas e alteradas ao longo do tempo e representar esse comportamento de forma estatística, numérica ou graficamente.

1.2 - Contexto e Objectivos

A monitorização e controlo de estruturas é uma tecnologia, que nos últimos 20 anos se tem desenvolvido largamente, por se apresentar cada vez mais como uma ferramenta que permitirá tanto a prolongação do tempo de vida das estruturas, assim como uma redução dos seus custos de manutenção. Este tipo de sistemas permite uma solução ao diagnóstico da deterioração das estruturas, e também um sistema de controlo activo que permite actuar sobre a estrutura em função das variáveis monitorizadas.

Pretende-se com a presente dissertação um estudo teórico, de sistemas de automação de supervisão e controlo, baseados numa arquitectura distribuída e de tempo real, bem como, das possíveis aplicações destes sistemas de automação em estruturas de engenharia civil, mais especificamente em pontes. Como os sistemas SCADA são parte integrante destes sistemas, este será outro dos pontos de estudo, no sentido de compreender a sua evolução histórica e arquitectura actualmente utilizada para os mesmos. Para obtenção ou manipulação dos dados dos níveis de controlo, relativamente aos níveis de processo são necessárias redes industriais, de campo, célula e fábrica, será realizado um estudo dos requisitos da comunicação necessária para cada um destes níveis, relativamente à largura de banda, velocidade de transmissão, tipo de tráfego, frequência de transmissão, entre outros.

As grandezas a monitorizadas na estrutura (ponte) serão extensões, acelerações/vibrações e temperatura. Será realizado um estudo relativo aos sensores passíveis de utilizar, para obtenção de cada uma destas variáveis, que serão monitorizadas, no caso das acelerações a uma frequência entre 100 e 200Hz e no caso das extensões e temperaturas a cerca de 1Hz.

Adicionalmente serão apresentados os possíveis actuadores para o sistema em causa, que serão actuadores de processo, como sinalizadores luminosos e amortecedores hidráulicos, que permitam eliminar a possível ressonância dos cabos de suporte da ponte; ou actuadores de informação, como alarmes, sendo estes a grande prioridade para análise do comportamento da estrutura. O sistema deve assim conseguir gerar alarmes, quando se verifiquem situações anómalas e deve ainda envia-los o mais rapidamente possível aos responsáveis pela obra.

Será implementada uma rede sem fios Wireless como rede de campo do sistema, por ser uma rede que satisfaz todos os requisitos de uma rede para esse nível e que permite reduzir bastante o custo e tempo de instalação do sistema de monitorização, por não ser necessária a utilização de cablagem excessiva.

Como rede de fábrica e de célula será utilizado um barramento CAN, que permite uma largura de banda suficiente para uma aplicação como esta, bem como transmissão de dados a longas distâncias, porque num sistema de monitorização de pontes podem existir centenas ou até milhares de sensores instalados ao longo de toda a estrutura separados por algumas dezenas, centenas ou até milhares de quilómetros.

Um dos principais objectivos desta dissertação é também o projecto, design e concepção de uma interface, que deverá permitir ao utilizador verificar o estado completo dessa estrutura e se achar necessário, actuar directamente sobre ela.

Por fim será implementado um sistema seguro de acesso remoto ao SCADA, via Web, escolhendo um servidor adequado bem como o desenvolvimento do seu código respectivo, e implementação de um sistema cliente para o browser.

Este projecto será realizado conjuntamente com o estudante João Miguel Figueiredo Ferreira, responsável pela parte de instrumentação - aquisição e condicionamento dos sinais dos sensores.

1.3 - Organização do documento

Este documento desenvolve-se em 8 capítulos distintos, o primeiro dos quais é a presente introdução, onde é realizado um enquadramento geral do tema a desenvolver, e se apresentam ainda os objectivos pretendidos com esta dissertação.

Seguidamente, no capítulo 2 é realizado um estudo teórico sobre sistema de supervisão e controlo em geral, bem como, a sua possível aplicação em estruturas de engenharia civil. Para além disto, são também, neste capítulo abordados os sistemas SCADA nomeadamente, o seu historial, a sua arquitectura e os seus constituintes

Por seguinte, uma das matérias principais na elaboração de qualquer sistema de engenharia, é uma percepção completa dos requisitos que este deve apresentar no seu aspecto final. Assim, no capítulo 3, é efectuado um levantamento dos requisitos para o sistema desenvolvido. Sendo em seguida apresentada a melhor solução para uma arquitectura do sistema, que é também neste capítulo apresentada, bem como, as escolhas das tecnologias utilizadas na sua implementação.

No quarto capítulo, é explicada a implementação da unidade central do sistema desenvolvido, que é implementada num computador que realiza praticamente todo o comando e controlo do sistema de monitorização. Desta unidade, fazem parte: a interface homem – máquina (HMI), onde será realizado todo esse processamento; a interface remota (Web Site) e ainda, um sistema de armazenamento de dados.

Posteriormente, no capítulo 5, será executada uma investigação relativa às tecnologias de comunicação, com ênfase sobre as tecnologias utilizadas no sistema desenvolvido: rede CAN-Bus e redes *wireless*. Sendo depois desta introdução teórica, explorada a implementação dos módulos de comunicação da rede CAN (*gateways* CAN – RS-232), seguida da implementação das unidades de processamento e comunicação intermédia do sistema, que transmitem e recebem dados, de e para as unidades de processo, utilizando uma rede sem fios.

O funcionamento das unidades de processo implementadas, que incorporam os sensores, os seus circuitos de aquisição e condicionamento de sinal e os actuadores do sistema, é compreendido no capítulo 6. Onde para além da implementação destes sensores, é realizado um estudo de algumas das grandezas estruturais de mais importante monitorização, incluindo os sensores utilizados para cada uma dessas grandezas, sendo que incide essencialmente sobre os sensores das grandezas mensuradas no sistema desenvolvido, tais como: extensómetros, acelerómetros e sensores de temperatura.

Por sua vez, no capítulo 7, são apresentados os resultados relativos às tecnologias utilizadas nos sistemas de monitorização, seguidos da sua discussão. Assim como, alguns resultados e sua apreciação, referentes a ensaios realizados no LABEST - Laboratório da Tecnologia do Betão e do Comportamento Estrutural, da Faculdade de Engenharia da Universidade do Porto, utilizando uma barra metálica sujeita a deformação e movimentos forçados.

Para terminar, no capítulo 8, será realizada em jeito de conclusão, uma síntese e apreciação crítica ao sistema desenvolvido e ainda, alguns desenvolvimentos futuros passíveis de serem realizados, como seguimento ao projecto que aqui se apresenta.

Capítulo 2

Sistemas de Supervisão e Controlo

2.1 - Sistema de Supervisão e Controlo de Estruturas

A sociedade moderna, actualmente depende de um conjunto extenso e complexo de infra-estruturas para manter a sua prosperidade económica e qualidade de vida dos cidadãos. Durante muito tempo, as infra-estruturas, essencialmente as de serviço de público, como pontes ou edifícios, sofreram de negligência e uso descontrolado, o que levava à sua rápida deterioração. Por estes factos, vive-se actualmente uma situação de crise estrutural iminente, impulsionada pelo elevado número de estruturas que necessitam de ser reforçadas, reabilitadas ou mesmo substituídas.

Foram vários os factores que conduziram a esta má condição das obras de engenharia civil, tais como os efeitos da corrosão em armaduras de betão armado, corrosão das estruturas metálicas, a própria deterioração por envelhecimento, o aumento dos seus requisitos estruturais ao longo do tempo e consequentemente o aumento de cargas que lhe são aplicadas, entre outros. As estruturas chegaram este estado, essencialmente devido à sua má inspecção e monitorização, que resulta numa constatação tardia destes problemas. Grande parte das estruturas, que sofreram com estas falhas dos responsáveis pelas obras, encontram-se em tal estado que o seu custo de reparação muitas vezes se aproxima do custo de substituição. Esta realidade, levou os Donos das Obras a procurarem novas tecnologias, como sistemas de supervisão e controlo estrutural (SHM - *Structural Health Monitoring*), esperando, que estes prolonguem o tempo de vida das estruturas e reduzam significativamente os seus custos de manutenção.

Nos últimos 20 anos, as tecnologias associadas a sistemas de monitorização e controlo de estruturas têm-se desenvolvido largamente, sendo esta uma área emergente na engenharia civil. Civiónica é a designação da área que combina a aplicação da electrónica às estruturas. A Civiónica baseia-se essencialmente na aplicação de sistemas de automação de supervisão e controlo recorrendo a componentes electrónicos, sensores e actuadores, para estudos dos comportamentos estruturais de obras de engenharia civil. Entenda-se então, por monitorização e controlo estrutural, o acesso a informação relativa ao comportamento de estruturas, utilizando técnicas de medição electrónicas e actuação inteligente no sentido de responder da melhor forma a um determinada situação. As estruturas supervisionadas e controlados de forma inteligente são conhecidas como estruturas inteligentes, tema que será aprofundado na secção 2.1.3.

A evolução dos sistemas de supervisão e controlo de estruturas, não ocorreu apenas pela necessidade de as monitorizar a longo prazo para uma melhor gestão da mesma, surgiu também devido à necessidade de avaliar o comportamento de novas soluções arquitectónicas e novos materiais utilizados em estruturas. Com o intuito de garantir, que mesmo não se tratando de estruturas anteriormente testadas, a sua segurança não está comprometida.

A instalação de sistemas de monitorização, é actualmente muito valorizada pelos responsáveis da exploração das obras, na medida em que fornecem dados essenciais aos modelos de decisão, que permitem tanto actuação instantânea sobre essas estruturas, como calendarizar e tipificar intervenções em estruturas existentes. Possibilitando deste modo, a sua reparação ou reabilitação, no caso de apresentar sinais evidentes de deterioração e/ou diminuição do seu desempenho do ponto de vista estrutural. Desta forma, os sistemas de monitorização são reconhecidos como ferramentas indispensáveis, para o incremento da eficácia das respostas, no âmbito do controlo de segurança de das estruturas, cujos níveis de exigência são progressivamente acrescidos, permitindo assim melhorar as correspondentes condições de utilização e aumentar a sua longevidade [1].

Esta monitorização, pode ser feita por sistemas de automação de supervisão e controlo, que agora, graças à evolução das tecnologias dos materiais e estruturas inteligentes, oferecem condições de supervisionar, controlar e actuar sobre estas, de forma inteligente.

2.1.1 - Sistemas de Automação Industriais

A automação industrial consiste no emprego de técnicas, softwares e/ou equipamentos específicos a um determinado equipamento ou processo industrial, para que a eficácia do mesmo seja aumentada. Tendo assim como principais finalidades: maximizar a produção; diminuir a emissão de resíduos de qualquer espécie; reduzir ao máximo o consumo de energia e/ou matérias-primas; melhorar as condições de segurança, quer seja material, humana ou das informações referentes a esse processo e ainda, reduzir o esforço ou a intervenção humana sobre o equipamento ou processo.

Por conseguinte, os sistemas de automação industrial, facultam uma importante contribuição adicional na conexão do sistema de supervisão e controlo com os sistemas corporativos de gestão das empresas. Esta ligação, possibilita que dados importantes da operação dos processos sejam partilhados, o que contribui para tornar mais ágeis os processos de decisão e mais confiáveis os dados que sustentam as decisões dentro da empresa, para deste modo melhorar a produtividade.

Um sistema de automação pode apresentar hierarquias de controlo estruturadas em vários níveis, apresentando em cada um desses níveis equipamentos e características distintas, permitindo assim uma estruturação hierárquica das funções do sistema.

O número de níveis hierárquicos não é um número fixo, normalmente encontra-se no intervalo de 4 a 6 níveis [2, 3]. A arquitectura, mais comummente, utilizada em indústrias de processo contínuo é de 5 níveis, dispostos de forma descendente da seguinte forma:

- Empresa - é efectuada uma gestão de alto nível, de planeamento estratégico e dos sistemas de informação;
- Fábricas - são realizadas tarefas de escalonamento da produção, monitorização de equipamentos e materiais;
- Controlo supervisório - é feito o controlo de cooperação entre as várias unidades que compõe o processo;

- Controlo regulatório - é realizado o controlo das operações de uma unidade do processo;
- Dispositivo - é o elemento de mais baixo nível do sistema, relativo aos sensores e actuadores, utilizados na indústria.

Numa indústria de automação é fundamental um fluxo de informação ilimitado e atempado, que tanto pode ser tanto como horizontal. No primeiro caso os dados são trocados entre equipamentos em níveis adjacentes. No caso de ser uma comunicação descendente, os dados enviados poderão ser comandos, ordens ou pedidos de níveis superiores a níveis inferiores. No caso de ser uma comunicação ascendente, as informações transferidas serão relativas à execução dos pedidos efectuados pelo nível superior, ou informação de situações anormais ocorridas, como por exemplo avarias. Já o fluxo de informação horizontal, ocorre entre equipamentos do mesmo nível, os dados trocados são relativos a comandos e informações para a cooperação entre os equipamentos do nível em causa. Para que estes fluxos de dados ocorram, é essencial a existência de uma infra-estrutura de tecnologias de informação especializada, com redes de comunicação individuais, que devem ser escolhidas e integradas de acordo com a necessidade de cada nível. Estas redes distinguem-se, entre si, em termos de funcionalidade e performance que fornecem aos dispositivos onde estão inseridas.

Como nos sistemas de automação existe uma enorme diversidade de dispositivos de campo (sensores e actuadores) e redes de comunicação industriais, é implicativa a utilização várias plataformas de software e linguagens de programação, para desenvolver tais aplicações. Fica assim evidente, a necessidade da integração de infra-estruturas de comunicação de uma indústria/fábrica, para realizar um fiável e atempado intercâmbio de dados [4].

Uma típica arquitectura de uma rede de comunicação em industrial de automação pode ser visualizada na figura 2.1.

Ao nível dos dispositivos de campo, são apenas trocados um conjunto pequeno e restrito de dados, relativos a monitorizações (sinais de sensores) e controlo de actuadores (motores, ferramentas, luzes, etc). A rede de campo actualmente mais utilizada é a *Profibus*, mas não é a única, outras soluções para redes de campo são, por exemplo as redes sem fios (802.15.4), redes CAN, ou redes baseadas em *Ethernet*. Estes dispositivos de campo são conectados a controladores do processo (autómatos programáveis, computadores, microprocessadores, etc.) que possuem a sua própria rede, no esquema (figura 2.1) representado como rede de controlo, mas que pode também ser chamada de rede de célula. Nesta rede, são trocados dados em tempo real, entre os vários controladores e/ou estações de controlo. Os dispositivos de nível de controlo servem tanto para controlo do processo, como para a sua supervisão. Neste nível, os dados trocados entre dispositivos contêm mais informação, que os da rede anterior. Uma das soluções, actualmente mais utilizada, para estas redes são as redes baseadas em Ethernet com pacotes TCP/IP.

No nível de gestão da empresa, é possível realizar o controlo da execução de processo/manufactura e gestão das várias aplicações da empresa. Para realizar essas tarefas é também necessária uma rede de comunicação com requisitos diferentes das anteriores, que se designa, normalmente de rede de fábrica, também tipicamente se utiliza uma rede *Ethernet* com pacotes TCP/IP. Este nível pode porventura ser acedido remotamente, via Internet.

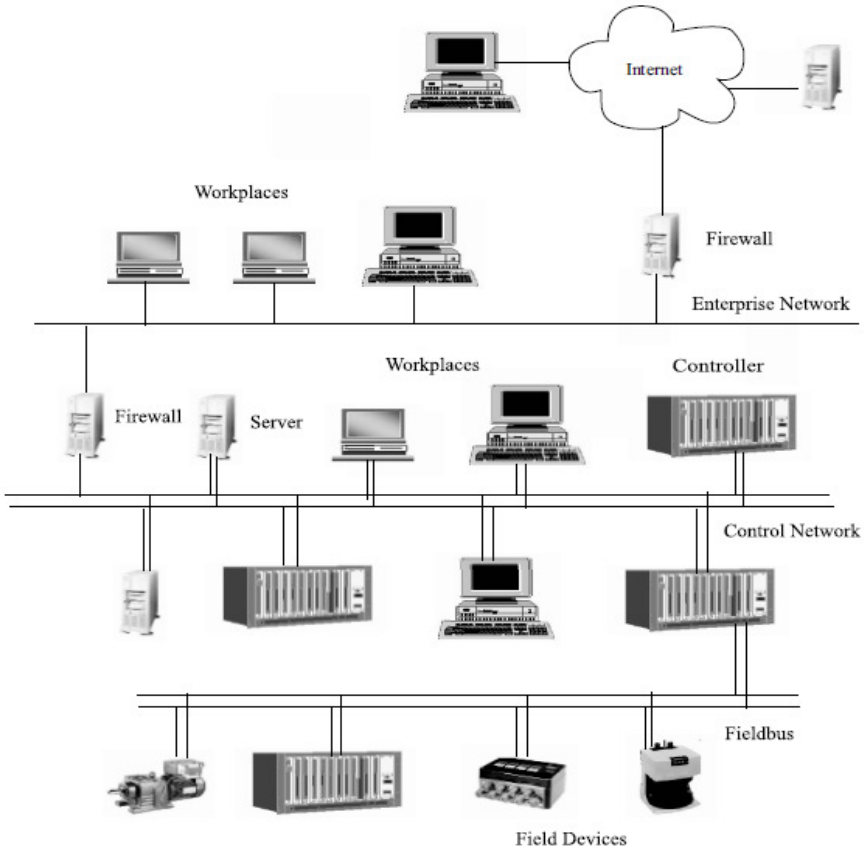


Figura 2.1 - Arquitectura de rede de comunicação para indústrias de Automação [4]

Dado numa indústria existir uma enorme variedade de processos, que devem ser monitorizados e controlados, é quase que obrigatório o projecto e implementação de sistemas de controlo e supervisão, que devem ser adaptados a cada tipo de indústria ou processo onde são integrados. Estes sistemas de controlo aplicam-se, tanto dentro do mesmo nível, como para níveis hierárquicos distintos. Dentro de um mesmo nível hierárquico o controlo pode ser [5]:

- Centralizado – em que o algoritmo de controlo é implementado num único dispositivo;
- Descentralizado – o algoritmo de controlo é executado num único dispositivo, mas a aquisição de dados do processo é realizada por dispositivos mais simples. Esta arquitectura já implica a utilização de redes de comunicação, que normalmente são ponto a ponto;
- Distribuído – onde o algoritmo de controlo é distribuído por diferentes dispositivos. Esta arquitectura implica a existência de redes de comunicação de maior complexidade que a anterior, como é o caso das redes de campo.

A interface programática entre os níveis de controlo de equipamentos e os de gestão da empresa, pode ser especificada segunda a norma IEC 62624, ou ISA 95, como é mais conhecida. Esta norma, foi desenvolvida para ser aplicada em qualquer tipo de processo utilizado nas fábricas, sejam elas de processos por lotes, processos contínuos ou processos repetitivos. Tem como principais objectivos: proporcionar uma terminologia consistente que será a base de comunicação entre o fornecedor e o fabricante; proporcionar modelos de

comunicação consistentes e fornecer modelos de operação sólidos que permitam simplificar a funcionalidade das aplicações e definir a informação a utilizar [6].

2.1.2 - Arquitectura de Sistemas Distribuídos e de Tempo Real

Sistema de Controlo Distribuído

Actualmente, utilizam-se cada vez mais arquitecturas computacionais de sistemas distribuídos com redes de comunicação, substituindo a primitiva organização de sistemas centralizados. A arquitectura distribuída tem uma implementação mais complexa, pois exige uma coordenação muito sofisticada entre os vários controladores, implicando um maior custo inicial na sua implementação, relativamente a outras soluções. No entanto, a relação qualidade-preço grande parte das vezes, essencialmente para soluções de média ou elevada complexidade, é compensatória. Com esta arquitectura, é possível ter um desempenho elevadíssimo do sistema, graças à execução dos algoritmos de controlo em paralelo, a um aumento da flexibilidade do sistema e uma maior tolerância a falhas do mesmo.

A capacidade de tolerar falhas por parte de um sistema de controlo distribuído é devida à utilização de unidades redundantes configuradas, como unidades tolerantes a falhas (FTU - Fault-Tolerant Unit). Esta característica, é uma vantagem tremenda para aplicações que necessitam de um elevado grau de segurança e confiabilidade, tais como aviões, carros, etc.

Um exemplo de aplicação de sistemas deste tipo é a aplicação em automóveis, de um controlador digital avançado, que lhe permite melhorar a estabilidade, performance, segurança, redução dos consumos de combustível, redução das emissões de dióxido de carbono, etc [7].

Estes sistemas de controlo, como já foi apresentado, são implementados de forma distribuída e comunicam entre si por uma ou mais redes de comunicação. As tarefas de controlo, sensorização, actuação ou processamento (algoritmo de controlo) são repartidas por diferentes módulos e interligadas pela rede de comunicação. Uma representação esquemática de um sistema de controlo distribuído pode ser visualizada na figura 2.2.

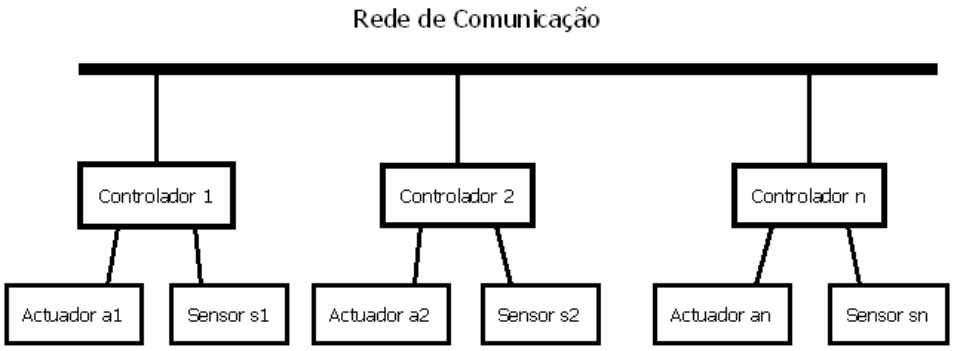


Figura 2.2 - Arquitectura de sistema de controlo distribuído

Um controlador *i*, com os seus os actuadores *ai* e os seus sensores *si* ilustrados no esquema anterior, representam uma unidade do processo.

Num sistema de controlo tradicional, com arquitectura centralizada, as tarefas de medição/sensorização, execução do algoritmo de controlo e actuação, são estritamente sequenciais. Num sistema distribuído, essas tarefas são executadas em paralelo ou de modo

sobreposto, introduzindo assim alguns problemas relativos à sincronização destas acções, que necessitam de ser correctamente tratados na fase de projecto do sistema [7].

Começa a ficar assim, cada vez mais notório a necessidade da utilização de um modelo de comunicação e cooperação, que garanta para além da sincronização das várias tarefas do sistema (sensorização, actuação e execução do algoritmo de controlo), também um correcto e controlado fluxo de dados entre os vários nós do sistema distribuído. Pois uma errada circulação destes dados terá reflexos negativos inevitáveis, no funcionamento e desempenho do sistema. Existem vários modelos de sincronização para tratar estes problemas, dos quais se destacam: o modelo Cliente-Servidor; o modelo Productor-Consumidor; o modelo Publicador-Subscritor e o modelo Mestre-Escravo.

O mais utilizado em sistemas distribuídos é o modelo Cliente-Servidor, aqui existem dois tipos de elementos: os servidores, que fornecem os serviços e os clientes de usam esses serviços (comunicação *unicast*). A comunicação neste modelo é iniciada sempre pelo cliente, invocando um serviço ao servidor, que apenas lhe responde a esse pedido. Os pedidos podem ser feitos síncrona ou assíncronamente, sendo esta última a mais comum. Ao utilizar uma comunicação assíncrona neste modelo, não é fácil estimar o atraso da comunicação provocado pelo tempo de resposta do servidor, por não se conhecer, à partida, a carga do servidor, ou seja, quantos clientes querem aceder a um servidor no mesmo instante. Por este motivo, este modelo é considerado temporalmente imprevisível.

O modelo Productor-Consumidor é constituído por dois componentes: os produtores, que produzem os serviços; e os consumidores que consomem os serviços. A comunicação é realizada por iniciativa do produtor, que publica um serviço que os consumidores podem consumir ou ignorar (comunicação *broadcast*). Este modelo é temporalmente previsível, pois o fluxo de dados é unidireccional, através de um conjunto de componentes, tornando a ordem de execução e o atraso ponta a ponta altamente previsível.

O modelo Publicador-Subscritor é parecido com o modelo anterior, o Publicador produz e publica os serviços por iniciativa própria e o Subscritor consome estes serviços. A diferença para com o modelo anterior, reside no facto de neste caso, o Subscritor ter que "subscrever" o conjunto de serviços oferecidos pelo Publicador que pretende consumir, evitando assim receber informação que não lhe interessa (comunicação *multicast*). Se o conjunto de serviços a que os subscritores se subscrevem estiver bem explícito, este modelo tal como o anterior torna-se temporalmente previsível.

No modelo Mestre-Escravo, fazem parte: os mestres que coordenam a execução das actividades e os escravos que executam os pedidos/ordens dos mestres, estas ordens podem ser de pedidos de informação/dados ou ordens de execução. A comunicação neste modelo é sempre iniciada pelo mestre, através do envio de pedidos (dados ou execução de tarefas) a um escravo, que lhe responde com os dados desejados ou com a execução das tarefas pretendidas (comunicação *unicast*). Neste modelo, como as funções são distribuídas pelos vários escravos torna-se simples estimar o atraso na realização de uma determinada tarefa, considerando-se assim este modelo temporalmente previsível.

O modelo Mestre-Escravo, será o modelo base utilizado no sistema que se pretende desenvolver, o modelo Produtor-Consumidor será também utilizado na rede CAN e o modelo Cliente-Servidor será utilizado para acesso remoto às variáveis do sistema, como ficará evidente em capítulos posteriores.

A utilização de um sistema distribuído de controlo exige, normalmente, características de controlo em tempo real, de forma a garantir que as operações executadas pelo sistema

terminam, antes do seu tempo máximo estabelecido. Estas características de tempo real são alargadas até às redes de comunicação, pois em sistemas deste tipo, o controlo da execução do código de dados, está distribuído em todos os nós que comunicam por estas redes e as restrições temporais, aplicam-se a todo o conjunto. Obrigando assim à utilização de protocolos de comunicação determinísticos, ou seja, com tempos máximos de transmissão previsíveis.

Sistemas de Controlo em Tempo Real

Genericamente um sistema de tempo real, é um sistema de controlo ou monitorização, que consegue acompanhar o estado de um dado processo físico e, se necessário, actuar a tempo sobre ele.

Designa-se por tempo real o ritmo de evolução (dinâmica) de um dado processo físico que não pode (ou não deve, no caso de simuladores) ser controlado externamente. Sendo assim, o meio envolvente é quem impõe ao sistema os requisitos temporais. Um sistema, só é de tempo real se conseguir actuar a atempadamente sobre esse meio envolvente, isto é, em consonância com o seu tempo real. Estes sistemas não necessitam de ser muito rápidos, devem é ser mais rápidos que o ritmo de evolução do seu meio envolvente.

Os sistemas de tempo real são normalmente sistemas embarcados, porque controlam um sistema genérico, do qual fazem parte [8]. O controlo computorizado de um automóvel, o controlo de um avião, o controlo de um robô, ou o controlo de produção numa fábrica, são alguns exemplos de sistemas de controlo em tempo real.

Os sistemas de tempo real podem ser classificados de dois tipos:

- Sistemas de tempo real críticos (*Hard Real-Time Systems*) – nestes sistemas o incumprimentos de uma restrição temporal pode resultar numa falha do sistema;
- Sistemas de tempo real não críticos (*Soft Real-Time Systems*) – aqui o não cumprimento dessas restrições, desde que não sejam em partes críticas do sistema, pode não ser sinónimo de uma falha no sistema.

Um sistema que controle de uma central nuclear, comparativamente a um sistema de base de dados de tempo real, é claramente um sistema de tempo real crítico, pois uma falha neste, poderá implicar uma catástrofe, pondo em risco vidas humanas.

No caso prático de monitorização de estruturas físicas, pode não ser possível utilizar um sistema totalmente de tempo real. Como o número de sensores instalados numa estrutura pode atingir valores muito elevados, até milhares, é fácil compreender que não existem processadores tão poderosos, que permitam fazer a monitorização atempada de tantos dados. A utilização de sensores inteligentes pode minimizar esta lacuna, estes serão apresentados na secção 2.1.4.

2.1.3 - Sistemas de Automação para Estruturas Inteligentes

Cada vez mais, os proprietários de obras de engenharia civil são confrontados com a necessidade da as monitorizar continuamente, com intuito de avaliar o seu desempenho e a sua integridade estrutural. Como já foi em cima analisado, uma solução para monitorização destas estruturas, pode ser conseguida por sistemas de automação de supervisão e controlo, os quais permitem para além dessa monitorização, também actuar de forma inteligente sobre estas, graças à evolução que se verificou nas tecnologias dos materiais, possibilitando assim, o aparecimento de estruturas inteligentes.

Uma estrutura inteligente consiste numa estrutura dotada de sensores, que lhes são embebidos como partes integrantes, e possivelmente por actuadores, que respondem em função de uma unidade de controlo dependente das informações recolhidas pelos sensores. Estas estruturas possuem a capacidade de agir e reagir de forma programada e inteligente, em função das solicitações e do meio ambiente. Desta forma, as estruturas inteligentes são dotados de um sistema de controlo activo, que lhes permitem, a partir de actuadores mecânicos especializados, que produzem forças ou deformações, actuar sobre as estruturas em função dos sistemas de monitorização.

A utilização de controlo activo em estruturas, tem sido reconhecido como uma das áreas de maior desafio na engenharia estrutural nos últimos anos, com especial destaque na área da dinâmica estrutural. Utilizando estes controladores, é possível modificar o comportamento da estrutura durante acções dinâmicas, como impactos, ventos ou sismos.

Soluções deste tipo trazem enormes vantagens, quer no aumento do tempo de vida útil da estrutura, quer na prevenção de danos estruturais, ou ainda na construção de estruturas cada vez mais arrojadas e atractivas. Por exemplo, através de controlo activo em pontes é possível limitar a deformação da estrutura num limite elástico e, portanto, prevenir danos estruturais e não estruturais. A prevenção de danos não estruturais, permite evitar o aparecimento de danos que no seu contexto, podem ser mais graves a longo prazo, do que propriamente danos estruturais, dos quais são exemplos, o aparecimento de fissuração, ou a corrosão das armaduras das estruturas [9].

Numa estrutura inteligente pode dimensionar-se um determinado número de membros activamente controlados, onde cada membro tem um sensor e/ou um actuador. O sensor mede por exemplo, os deslocamentos dos graus de liberdade e o actuador aplica a força necessária para correcção apropriada, como resposta a essas medições. Um outro exemplo, será para ponte suspensas por cabos, quando se verifica pelos sensores de acelerações/vibrações (acelerómetros) que esses cabos estão em alta ressonância, pode ser activado um amortecedor capaz de a eliminar. Todo este controlo é realizado por um sistema de monitorização em malha fechada, que garante um permanente *feedback* entre os sensores, computador central e os actuadores. O controlo activo das estruturas pode ser realizado de duas formas: uma delas em que a estrutura reage quando uma grandeza de entrada ultrapassa certos limites, superiores ou inferiores, previamente estabelecidos – controlo por histerese; outra, em que as grandezas de entrada são processadas em controladores de níveis hierárquicos superiores do sistema de supervisão e controlo, para posteriormente, ser formulada uma resposta adequada na estrutura.

Será seguidamente realizada uma pequena abordagem a uma arquitectura possível, para um sistema de supervisão e controlo, aplicável ao conceito de estruturas inteligentes.

Arquitectura de um Sistema de Supervisão e Controlo para Estruturas Inteligentes

Na figura 2.3, está apresentada uma possível arquitectura hierárquica para um sistema de supervisão e controlo aplicável a estruturas inteligentes.

Nessa figura *y* representa o conjunto de dados analógicos recolhidos pelos sensores directamente para o computador periférico (unidade de controlo activo) *w* representa o vector de dados analógicos que transitam directamente para o computador central e *ui* representa as instruções enviadas pelos elementos de controlo para os actuadores, sendo u1 e u2 originários de unidades de controlo de níveis hierárquicos distintos. A primeira é originária

da unidade de controlo de mais baixo nível, a unidade de controlo activa e a segunda, proveniente de uma unidade de controlo de mais alto nível, referente à unidade de gestão do processo, representada no esquema como unidade e comando e controlo.

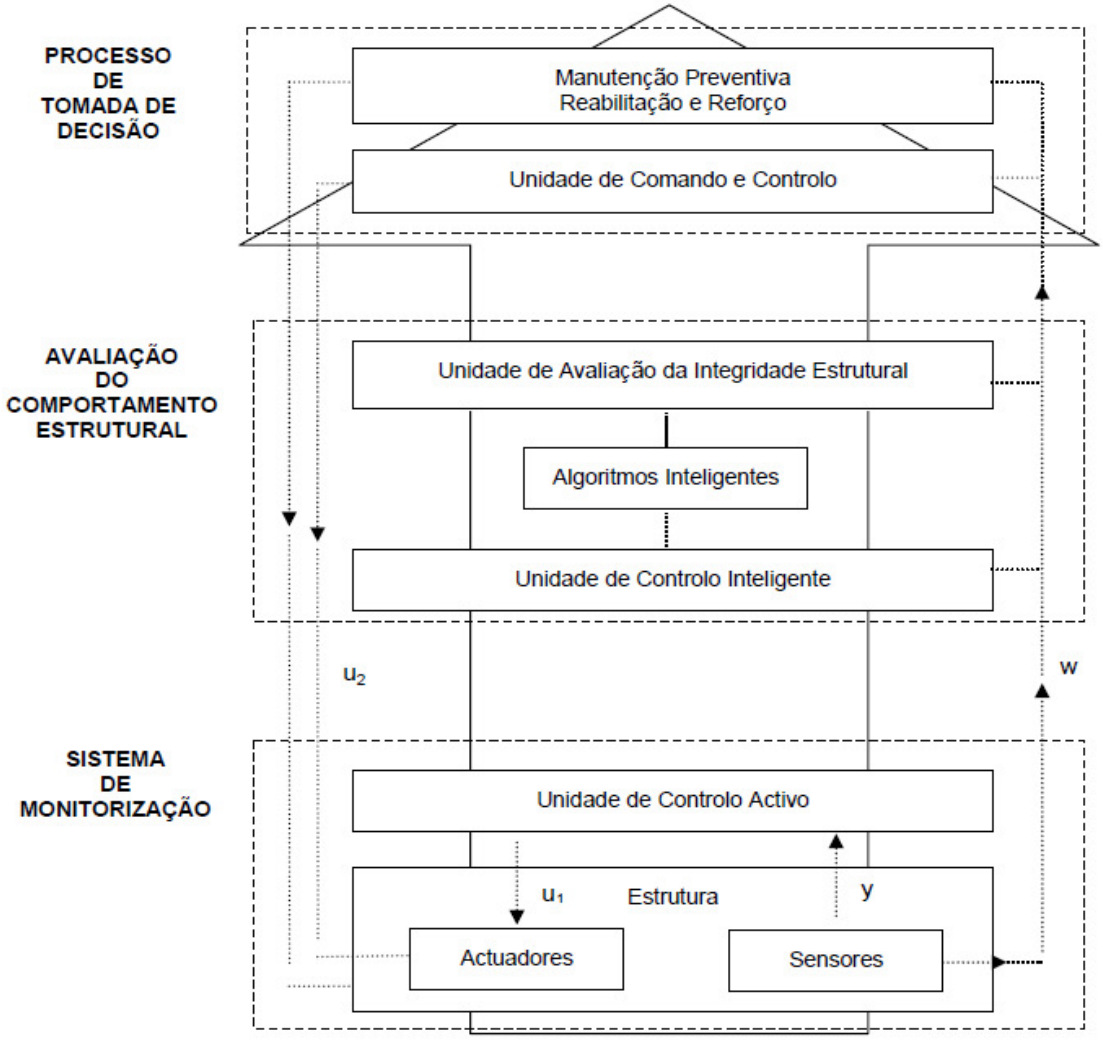


Figura 2.3 - Arquitectura de um Sistema de Supervisão e Controlo para Estruturas Inteligentes [9]

O elemento chave de uma monitorização deste tipo, reside no tratamento inteligente de dados num computador central, relativos à avaliação do comportamento estrutural, através dos chamados algoritmos inteligentes. Estes algoritmos, tratam os dados com dois objectivos bem definidos ao nível de duas unidades: (i) unidade de controlo inteligente; e (ii) unidade de avaliação da integridade estrutural.

- Ao nível da unidade de controlo inteligente (i), os algoritmos devem ser capazes de avaliar comportamentos, prever respostas e propor ajustes estruturais à unidade de comando e controlo;
- Ao nível da avaliação da integridade estrutural (ii), os algoritmos devem ser capazes de identificar danos estruturais para que, em função do tipo de danos, se executem as operações de manutenção, reabilitação ou reforço adequadas.

Relação entre Arquitectura de Sistemas de Monitorização de Estruturas Inteligentes e Arquitectura de Indústrias de Processos Contínuos

No inicio deste capítulo, foi apresentada uma possível arquitectura de 5 níveis para um processo de produção contínuo. Esses níveis são: empresa, fábrica, controlo supervisório, controlo regulatório e dispositivo. Sendo a empresa o mais alto nível e o dispositivo o nível mais baixo desta hierarquia.

Empresa, elemento de mais alto nível da arquitectura de uma indústria de processo contínuo, é responsável pelo planeamento estratégico de produção da empresa. Este nível na arquitectura de sistemas de monitorização de estruturas, pode ser representado pela unidade de comando e controlo, que realiza a gestão de alto nível do processo global, podendo enviar comandos aos actuadores do processo.

A fábrica da indústria de automação está representada na arquitectura das estruturas pelas unidades de avaliação e de controlo inteligente. No nível da fábrica são realizadas as tarefas de escalonamento da produção e monitorização dos materiais utilizados, que são tarefas semelhantes às realizadas nas unidades que as representam nesta nova arquitectura, apresentadas em cima, de avaliação de comportamentos do processo e apresentação de propostas de actuação ao nível superior.

O nível de controlo supervisório é responsável pelo controlo de coordenação entre as várias unidades que compõe o processo, ou seja, pelas unidades de controlo activo e dos dispositivos de campo (sensores e actuadores). Este nível, está omisso na arquitectura para estruturas inteligentes representada na Figura 2.3, pois, nesta apenas se encontra representada uma unidade de processo. No entanto, este é um nível essencial numa arquitectura de sistemas distribuídos. E numa arquitectura mais completa de um sistema de monitorização de estruturas onde existissem mais unidades de controlo activo, este nível estaria representado, nesta arquitectura, imediatamente acima dessas várias unidades.

No controlo regulatório, é feito o controlo directo das operações de uma unidade do processo. Este controlo é equivalente ao representado na arquitectura de estruturas inteligentes, como unidade de controlo activo.

Por fim, está representado na Figura 2.3, como “Estrutura” o nível mais inferior desta arquitectura, relativo aos sensores e actuadores do processo. Este nível, é representativo do nível dos dispositivos da arquitectura de um processo contínuo.

2.1.4 - Rede de Sensores Inteligentes

Os sensores, são componentes utilizados em vários dispositivos e sistemas, que fornecem informação sobre os parâmetros que se pretendam mensurar, ou para identificar estados de controlo do sistema. Em sistemas de supervisão e controlo, é essencial uma escolha apropriada dos tipos de sensores a utilizar, de acordo com as potencialidades dos sensores e dos objectivos que se pretendem obter de cada sensor, para que forneçam a informação pretendida com a maior precisão possível.

Com o auxílio de microprocessadores, é possível tornar sensores dotados de inteligência. Este facto, só foi praticável a partir da migração verificada nos últimos anos das redes de comunicação, para a comunidade dos sensores que tornou exequível, a utilização dos sensores inteligentes, que transmitem as suas medições directamente, a um qualquer instrumento ou sistema de supervisão. Ou seja, consegue-se assim descentralizar a inteligência e o controlo dos dispositivos de controlo, para os níveis dos transdutores (sensores e actuadores).

Utilizando sensores inteligentes, consegue-se mais flexibilidade no sistema, melhorar as suas performances, maior facilidade da sua instalação, actualização e manutenção. Por todas estas vantagens, cada vez mais, ao nível industrial, são utilizados este tipo de soluções, de uma arquitectura de controlo distribuída com sensores inteligentes [4].

Para aplicações de supervisão de estruturas físicas, é necessário realizar um estudo cuidadoso sobre o número e localização dos sensores na estrutura em causa. Pois em aplicações deste tipo, pretende-se detectar qualquer tipo de dano estrutural antes que este seja considerado crítico. Por outro lado, a aplicação de sensores numa estrutura, não deve introduzir qualquer alteração no comportamento da mesma, assim a presença de sensores, de cabos de ligação entre eles, postos de observação e outros acessórios necessários, devem ser considerados na fase do projecto destes sistemas.

Redes de sensores sem fios e redes de sensores de fibra óptica são duas possíveis soluções para redes de sensores inteligentes, que poderão ser aplicadas em estruturas (pontes, edifícios, etc.).

As redes de sensores de fibra óptica têm vindo cada vez mais a ser utilizadas em sistemas de monitorização, graças às suas inúmeras vantagens como: imunidade a interferências electromagnéticas; capacidade transmissões de sinais a longas distâncias; dimensão e peso reduzido; elevada resistência à corrosão e água; entre outros [10]. São no entanto redes muito caras devido ao elevadíssimo custo da fibra óptica e dos seus elementos de ligação.

A tecnologia de redes de sensores sem fios está ainda em fase de estudo e desenvolvimento, mas espera-se que se torne uma alternativa a curto prazo, pois apesar de serem mais lentas e menos confiáveis, prevê-se que com a utilização de redes sem fios se reduza substancialmente o custo de sistemas de monitorização estrutural. Dado que, em aplicações convencionais destes sistemas, 25% do seu custo total e 75% do tempo de instalação, se devem à instalação dos cabos [10]. Esta será, a rede de sensores utilizada no sistema em estudo.

A estrutura destas redes de sensores sem fios, consiste em dispositivos autónomos, que possuem os sensores acoplados, espalhados pela rede, normalmente designados por nós-sensores. Estes dispositivos, têm capacidade de comunicar entre si e com estações locais, que armazenam os valores medidos. Posteriormente, estes valores serão enviados a uma estação central, o *gateway*. Este *gateway*, colecta todos os dados medidos em cada nó e envia-os para um computador, através de uma conexão, que poderá ser por exemplo, do tipo Ethernet ou USB [11]. Uma outra possibilidade, mais barata, mas menos robusta, é a conexão por RS-232, que será a utilizada neste sistema.

2.1.5 - Sistemas de Aquisição

Existem variados tipos de sistemas de aquisição, com diversas funcionalidades. Pode-se, no entanto afirmar que o sistema de aquisição é responsável pela recepção, condicionamento, armazenamento e transmissão das leituras registadas nas redes de sensores. Com efeito, para poder processar, armazenar e transmitir os dados adquiridos pelos diversos sensores é necessário numa primeira fase, amplificar o sinal recebido e converter este de analógico para digital (A/D). A figura 2.4, esquematiza as diferentes fases de um sistema de aquisição, desde que o sinal é adquirido, até ao seu processamento final.

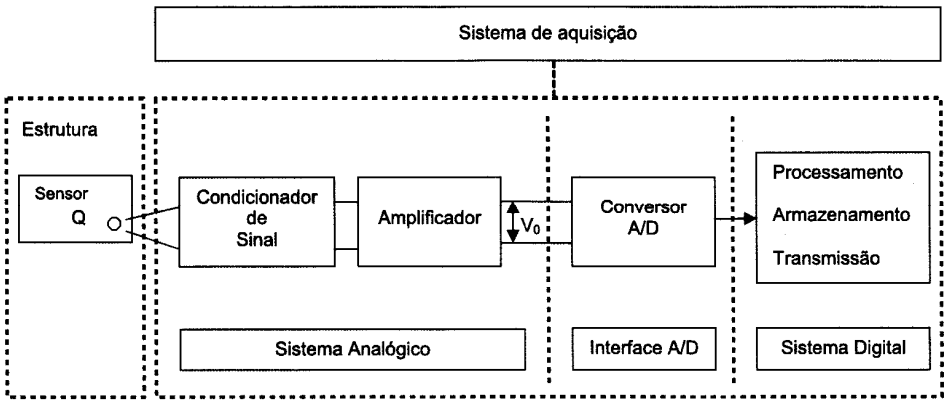


Figura 2.4 - Esquema de um Sistema de Aquisição [12]

Os sistemas de aquisição, podem ser constituídos por placas de aquisição integradas em computadores, ou serem equipamentos de funcionamento autónomo dispensando ligação permanente a estes. Os primeiros, são em geral equipamentos mais económicos e potentes. No entanto, como desvantagens apresentam o facto de serem sistemas de monitorização centralizados; a necessidade de disporem de cabos de ligação aos sensores de grande comprimento; e ainda a necessidade de existência de um computador na obra. Os segundos sistemas são em geral, menos potentes e permitem interrogar um número inferior de sensores, o que os torna mais dispendiosos. Contudo, sendo dotados de ligação em rede, permitem uma instrumentação distribuída na estrutura, com uma redução significativa do comprimento dos cabos de ligação, e por consequência, das perdas que lhe estão associadas, assim como a possibilidade de serem facilmente instalados em caixas ou armários de protecção.

Na figura 2.5a, apresenta-se um posto de observação, no qual estão integrados dois equipamentos de aquisição autónomos, da série DT515 da Data Taker, utilizados para a aquisição de dados a partir de sensores eléctricos. Estes aparelhos, constituem uma boa solução para a monitorização estrutural sob acções de média/baixa frequência. A sua versatilidade e robustez, têm sido demonstradas na possibilidade de centralizar num único equipamento a capacidade de interrogar os sensores e armazenar as leituras.

Por sua vez, na figura 2.5b demonstra-se um sistema de aquisição do tipo modular desenvolvido pela National Instruments, o CompactRio. Este, foi desenvolvido para sistemas que exijam um alto desempenho e fiabilidade. Possui memória interna e é capaz de adquirir sinais analógicos ou digitais de diversos tipos de sensores: tensão; corrente; temperatura; acelerómetros; extensómetros; etc. Adapta-se às diferentes necessidades de cada sensor, em termos de taxa de amostragem, amplitude do sinal, pré-processamento, entre outros e é capaz de, apesar destas diferenças, adquirir simultaneamente dados de diferentes sensores.

Os sistemas de aquisição de dados da National Instruments, são mais robustos, flexíveis, de menores dimensões e que permite adquirir dados a frequências mais elevadas que os da Data Taker, o que o torna ideal não só para ensaios estáticos, mas também para dinâmicos. A sua programação é realizada em LabVIEW, que é actualmente uma ferramenta bastante utilizada, por ser muito completa e de fácil manuseamento. Mas é por isto uma ferramenta de utilização mais difícil que a anterior para o operador, que exige maiores conhecimentos da mesma, para o tratamento de informação dos módulos de aquisição distintos.

a) b)

Figura 2.5 - Sistema de Aquisição: a) DT515 da *Data Taker* [13] e b) *CompactRio* da *National Instruments [10]*

2.1.6 - Segurança e Consistência em Sistemas de Automação Distribuídos

A possibilidade de acesso remoto ao sistema de automação expõe-no a ataques externos, que podem facilmente comprometer a integridade do sistema e pôr em perigo, o seu correcto funcionamento.

A segurança de um sistema computacional, pode ser definida pela sua capacidade de prevenir e detectar acções não autorizadas. É necessário em sistemas seguros garantir:

- Integridade, impedindo modificações de informação não autorizada;
- Confidencialidade, impedindo o acesso não autorizado à informação;
- Disponibilidade, permitindo o acesso autorizado à informação.

Um dos primeiros aspectos a ter em conta no estudo da segurança em sistemas distribuídos é a distinção entre políticas de segurança e mecanismo de segurança [14].

Para implementar uma política de segurança é em primeiro lugar, necessária a realização de uma análise específica das possíveis ameaças do sistema, ou seja, para definir quais as acções autorizadas ou não autorizadas e a que utilizadores (autenticações). Esta é implementada, recorrendo a mecanismos de segurança. Sendo esta implementação realizada a partir de mecanismos de segurança.

Na concepção de um sistema de segurança, há um conjunto de aspectos a ter em conta. O primeiro aspecto diz respeito à escolha da camada onde se devem implementar estes mecanismos de segurança (nos sistemas de comunicação, na aplicação, etc.). Em seguida estudar o sistema e perceber se este deve ser mais funcional, ou seja mais simples, ou se deve garantir um grau de confiança elevado, implicativo de um sistema mais complexo. Por fim, é necessário perceber de que componentes depende a segurança do sistema, para escolher se a concepção de mecanismos de segurança deve ser centralizada ou descentralizada.

Um mecanismo de segurança, mais utilizado, para impedir o acesso à informação relevante a utilizadores não autenticados, isto é, que permite o acesso apenas a utilizadores

devidamente autenticados, é o mecanismo de criptografia, baseado em algoritmos de encriptação/desencriptação. Deste mecanismo, existem dois tipos distintos:

- Chave Partilhada, é mecanismo que cifra as suas mensagens com a mesma chave que as decifra, e apenas as entidades autorizadas têm acesso a essa chave;
- Chave Pública, é um mecanismo que utiliza uma chave para cifrar do tipo pública, que todas as entidades podem ter acesso, mas para decifrar utiliza uma chave privada, diferente da chave utilizada para cifrar, a que apenas entidades autorizadas têm acesso.

Um exemplo de mecanismo de segurança que pode ser utilizado em sistemas de autenticação, é o algoritmo de encriptação de chave pública é o MD5 (*Message-Digest algorithm 5*). Este é um algoritmo de *hash* de 128 bits, unidireccional, utilizado na verificação da integridade de arquivos e logins. Algoritmo de *hash,* significa que são gerados, por um algoritmo próprio, um conjunto de bits, neste caso 128, a partir dos dados originais, que são visualizados no formato de números hexadecimais. A segurança na utilização deste método, deve-se ao facto de ser um algoritmo unidireccional, ou seja, uma trama *hash* MD5 depois de criada não pode ser novamente convertida no seu valor original. A entidade que irá verificar a integridade desta trama, compara-a com uma outra trama MD5 com mensagem confiável, que só ela possui e apenas se essa comparação for bem sucedida o utilizador é autenticado.

Ao nível das redes de campo, normalmente não existem grandes recursos de segurança, pois estas estão geralmente situados em instalações, que requerem acesso autorizado, e a eliminação ou deturpação de informação destas redes, implica um acesso ao seu meio.

Verificação da Consistência de Dados Trocados – CRC

Num sistema onde são utilizadas comunicações com pacotes, é necessária uma verificação da consistência da mensagem/pacote, enviada face à mensagem recebida. Esta, pode ser deturpada por interferências externas, electromagnéticas por exemplo. Detectar e possivelmente corrigir esses erros, é função da camada de ligação de dados do modelo OSI, apresentado mais à frente. Como essa camada não existe em todas as redes de comunicação, como por exemplo nas redes de fibra óptica, nas redes de comunicação série, RS-232 ou RS-485 que apenas implementam a camada física do modelo, é necessário implementar métodos capazes de detectar esses erros que possam ocorrer na comunicação.

A forma de detecção de erros mais eficaz, numa troca de dados série, com o mínimo *hardware* possível é a partir do *Cyclic Redundancy Check* (CRC). [15]

O CRC é, tal como MD5, um algoritmo de *hash* que gera, a partir de um conjunto de dados, pacote de uma mensagem por exemplo, um valor de poucos bits que permite detectar a ocorrência de erros na transmissão ou armazenamento da mensagem.

O método de verificação, consiste em calcular o valor do CRC de um determinado pacote, do lado do transmissor, que lhe será anexo. Após a transmissão esse valor é verificado pelo receptor, que calcula novamente o CRC da mensagem e o compara com o valor anexado, se forem iguais significa, com bastante certeza, que a mensagem foi trocada correctamente.

Os códigos de CRC podem ser considerados como códigos polinomiais, isto é, por exemplo o valor binário 000101101_2 pode também ser como um polinómio da seguinte forma:
$B(x) = 0x^8 + 0x^7 + 0x^6 + 1x^5 + 0x^4 + 1x^3 + 1x^2 + 0x^1 + 1x^0$, ou seja, $B(x) = x^5 + x^3 + x^2 + 1$.

A fórmula de cálculo do CRC, baseia-se numa divisão aplicada aos números binários. No entanto, nesta divisão em vez de realizar subtracções sucessivas, como numa divisão comum, realiza-se a operação lógica XOR sucessivamente.

Para calcular então o conjunto *R*, o CRC com *r* bits dados, de um conjunto de *D* com *d* bits de dados, recorre-se a um outro conjunto de bits *G*, denominado de gerador com *r*+1 bits de dados, cujo bit mais significativo tem obrigatoriamente que ser 1. Este gerador tem de ser comum para todos os elementos de uma rede que possam emitir ou receber mensagens, pois o cálculo do CRC do lado do emissor é realizado recorrendo a este valor e a verificação do lado do emissor será realizada com a mesma grandeza. Este método é análogo a um algoritmo de encriptação de chave partilhada.

A fórmula de cálculo do CRC é então a seguinte: $R = resto[\ D \times 2^r\ /\ G\ ]$, não esquecendo que na divisão utilizam-se XOR sucessivos, ao invés de subtracções.

Suponha-se então, que se pretende enviar o conjunto de dados 111100101_2, com um gerador 101101_2 ou $G(x) = x^5 + x^3 + x^2 + 1$, o que implica um grau de gerador (*r*) de 5, logo realizando a primeira operação da formula ($D \times 2^r$) obtém-se 11110010100000_2, faltando agora a divisão desse valor pelo gerador. Essa divisão está representada na figura 2.6. Neste caso, obteve-se pelo cálculo do CRC 1010_2. O valor obtido, de 4 bits seria sempre um valor com um número de bits igual ou inferior a *r*.

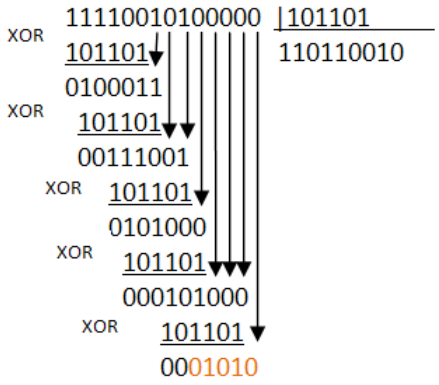


Figura 2.6 - Exemplo cálculo de CRC

As formas de cálculo de CRC mais usualmente utilizadas em comunicações digitais de dados, incluem o CRC-16 e o CRC-16-CCIT, em que cada um deles, origina um valor de CRC de 16 bits. [15]

Existem no entanto, outras formas de cálculo de CRC padronizadas, entre as quais os CRC-15-CAN de 15 bits, utilizado em comunicações CAN com o seguinte polinómio: $CRC\text{-}16\text{-}CAN(x) = x^{15} + x^{14} + x^{10} + x^8 + x^7 + x^4 + x^3 + 1$ e o CRC-8-Dallas/Maxim, um CRC de 8 bits utilizado em microprocessadores da Maxim para troca de dados com a memória ROM, que utiliza o polinómio: $CRC\text{-}8\text{-}Dallas/Maxim(x) = x^8 + x^5 + x^4 + 1$.

2.2 - SCADA

Um SCADA (*Supervisory Control and Data Acquisition*) é um software que permite monitorizar e controlar um processo, pelo acesso remoto a dados desse processo, que são compilados e posteriormente apresentados ao operador. Possibilita ainda, através de ferramentas de comunicação específicas a cada caso, controlar o mesmo. Para além disso, estes sistemas facultam uma interface entre os níveis de controlo do sistema e os níveis superiores de gestão.

Um sistema SCADA deve respeitar os seguintes objectivos [16]:

- Funcionar em qualquer computador standard, com sistema operativo *Windows*;
- Arquitectura aberta que permita a sua combinação com aplicações standard, possibilitando ao programador a criação de soluções de gestão e supervisão optimizadas;
- Instalação simples, sem grandes exigências a nível do hardware, fáceis de utilizar e com interfaces simples para o utilizador;
- Permita a integração com ferramentas de gestão e produção;
- Fácil de reconfigurar, permitindo uma adaptação ou crescimento a mudanças na empresa;
- Ser independente do ramo de actividade e da tecnologia utilizada;
- Conter integradas funções de gestão e supervisão;
- Comunicações flexíveis e transparentes, que permitam que o operador comunique de forma simples com a equipa da fábrica e com a restante empresa, a partir de redes de locais e de gestão.

2.2.1 – Historial dos sistemas SCADA

Um dos primeiros sistemas de telemetria, que consiste na medição de dados remotamente, foi utilizado em sistemas ferroviários, para controlar a aproximação de comboios de uma bifurcação, e seleccionar/controlar, o caminho que estes deveriam seguir. As redes de comunicação destes sistemas eram cabladas e utilizavam *switchs* electrónicos junto das linhas que permitiam o envio de informações para uma estação central. Estas informações eram visualizadas em sinais luminosos que indicavam a proximidade de um comboio e o estado da bifurcação, podendo assim, os operados alterar o estado da bifurcação, se necessário. Foram sistemas de telemetria como estes que deram origem a sistemas SCADA.

Este tipo de soluções era uma boa escolha para sistema fixos (sem mobilidade), como era o caso, mas não se podiam utilizar para sistemas onde a estação de recolha de dados era móvel.

Nos anos 20, com o desenvolvimento da indústria aeronáutica, tornava-se cada vez mais necessária, a aquisição de dados, para sistemas deste género, ou seja, para sistemas móveis. Por exemplo, os voos iniciais de aviões e foguetões normalmente acabam em despenhamentos ou em locais indesejados, pelo que colocar humanos para realizar estes testes, era difícil e perigoso.

Por esta altura, ocorria uma significativa evolução das tecnologias de transmissão de dados por frequência de rádio, que cada vez se tornavam mais fiáveis e permitiam uma transmissão de quantidades de dados, apresentavam-se assim como uma solução possível para este problema. No entanto, esta tecnologia continuava a ter uma enorme limitação, por apenas permitir uma transmissão de dados unidireccional. Assim, apenas era possível realizar uma recolha de dados, não se conseguia actuar/controlar o sistema [17].

As comunicações por fios sofriam, ao mesmo tempo que as tecnologias anteriores, uma enorme evolução. Conduzindo assim a melhorias muito significativas da fiabilidade, destas comunicações e introduzindo o conceito de comunicações bidireccionais, possibilitando assim efectuar, numa mesma rede, não só a recolha de dados como a execução de funções de

controlo. Ficava apenas a faltar a possibilidade de utilizar comunicações rádios (sem fios) e de realizar também uma comunicação bidireccional.

Por volta dos anos 60, apareciam os primeiros computadores digitais, que permitiram uma enorme flexibilidade para sistemas de telemetria, essencialmente nas estações de recolha de dados, permitindo-lhes centralizar todos os dados recolhidos no terreno, o que tornava estes sistemas mais simples e eficazes. No entanto, todos os problemas de automatização de processos em indústrias, eram da responsabilidade de cada fabricante. Cada um deveria resolver os seus problemas por si mesmo, desenvolvendo elementos electrónicos específicos para cada solução. Normalmente, estes sistemas tinham uma quantidade de memória reduzida pelo que necessitavam de comunicar constantemente com os seus sistemas de controlo para lhes enviar os dados e utilizavam normalmente, linguagens de programação, pouco conhecidas.

Ainda nos anos 60, fabricantes de sistemas electrónicos como a Siemens, Square-D ou a Allen-Bradley apareceram com uma nova geração processadores/autómatos, capazes de controlar uma enorme quantidade de entradas e saídas. Tratavam-se de controladores desenhados para ambientes muito severos, como é o caso dos ambientes industriais, eram no entanto, muito grandes e pesados e para além disso muito caros. Graças à constante evolução dos componentes electrónicos, que são cada vez mais baratos e pequenos, origina-se uma constante redução do tamanho, peso e custo, destes autómatos. Como resultados, apareceram os micro-PLCs (autómatos programáveis), por volta dos anos 80. Que permitiam um controlo modular, adaptando-se às necessidades das fábricas e que utilizam linguagens de programação genéricas, tornam-se num êxito imediato em ambientes industriais [16].

A designação de SCADA, para os sistemas de telemetria e controlo, foi utilizada pela primeira vez no inicio dos anos 70. Nesta década, verificava-se já uma grande evolução em tecnologias de rádio, que iam cada vez mais, substituindo as comunicações cabladas.

A evolução constante dos computadores originou por si só, uma previsível evolução também para os sistemas SCADA. Verificam-se melhorias significativas, quer ao nível das interfaces com o operador, quer nos dados que fornecem, podendo oferecer informações estatísticas por valores numéricos ou por gráficos. Como se verificavam que os computadores estavam cada vez mais baratos, deixava de ser necessária a utilização de uma arquitectura de sistemas SCADA centralizada, podendo optar-se por arquitecturas distribuídas [17].

Também por esta altura, as redes industriais e redes de campo, eram cada vez mais eficazes, permitindo uma recolha de dados muito robusta, bem como a execução de funções de controlo cada vez mais fiável, assim, estas trocas de dados realizam-se a velocidades cada vez maiores. Passam ainda de ser menos utilizadas as comunicações de rádio, apostando-se cada vez mais noutros tipos de comunicações sem fios, como comunicações por satélite, *bluetooth* ou por *wireless*.

Desde então, até aos dias de hoje, foram sendo acrescentadas cada vez mais funcionalidades aos sistemas SCADA, como: a inclusão de bases de dados, quer locais quer remotas, permitindo um registo de históricos mais eficazes e completo; possibilidade de acesso simultâneo das funcionalidades do sistema de diversos operadores do sistema; desenvolver estas tecnologias na Web, permitindo o acesso aos SCADAS remotamente, em qualquer hora e local, para utilizadores devidamente autenticados, implicando assim a necessidade de se implementarem mecanismos de segurança adequados, para não comprometer o bom funcionamento do sistema, por possíveis ataques externos.

2.2.2 - Arquitectura de sistemas SCADA

Como já vem sendo referido, os SCADAS são sistemas com capacidade para fornecer a um operador o estado de um determinado sistema e ainda a possibilidade de executar funções de controlo sobre o mesmo. Estes SCADAS, são uma das principais partes integrantes dos sistemas de supervisão e controlo, utilizados em indústrias de automação, dos quais também fazem parte os dispositivos de campo (sensores e actuadores), bem como os sistemas de comunicação (redes industriais e de redes de campo). Uma estrutura básica de um sistema de supervisão e controlo, pode ser visualizado no seguinte esquema (figura 2.7).

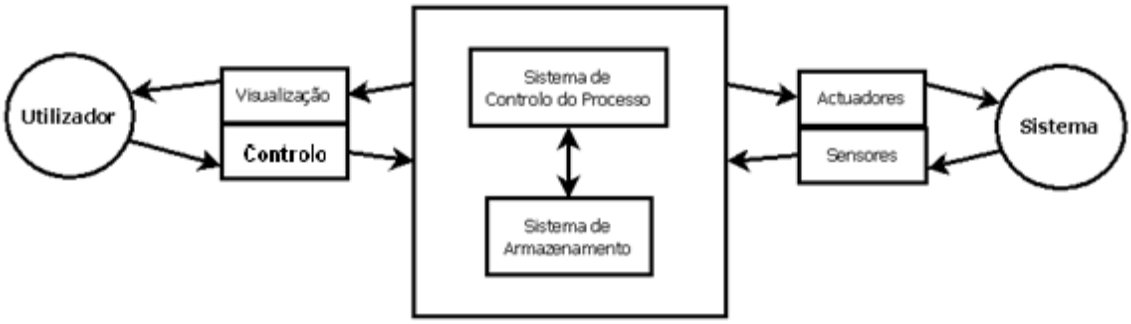


Figura 2.7 - Estrutura básica de um sistema de supervisão e controlo

O utilizador, a partir de ferramentas específicas, de visualização e controlo, tem acesso ao sistema de controlo do processo, sendo a comunicação, entre estes dois elementos, normalmente feita recorrendo a redes de comunicação industriais, que podem ser, por exemplo, baseadas em CAN-Bus, ou Ethernet. Esta ferramenta pode no entanto ser uma ferramenta Web visualizada, por exemplo, através de um *browser* em que os dados serão trocados utilizando pacotes UDP ou TCP/IP.

O sistema de controlo do processo, unidade central do sistema SCADA (*MTU, Master Terminal Unit*), consegue visualizar o estado do sistema/processo, recolhendo os dados dos sensores de campo e mostra-os ao utilizador, a partir de uma interface homem-máquina (HMI). Baseando-se em comandos recebidos do usuário, o sistema de controlo do processo executa as acções de controlo pelos actuadores. Esta unidade central, pode ainda guardar todas as informações geradas da supervisão e controlo em sistemas de armazenamento, permitindo a sua posterior análise. Por vezes o MTU e o HMI são desenvolvidos no mesmo computador e no mesmo programa, não havendo assim necessidade de trocar informações entre estes elementos. Isto pode ser conseguido se a linguagem de escrita do programa for uma linguagem gráfica, como são casos Java, C#, FreePascal, Visual C++, Visual Basic, etc…

O sistema SCADA interage com os dispositivos de campo a partir de unidades remotas conhecidas por RTU (*Remote Terminal Unit*), que são normalmente implementados em autómatos programáveis, ou microprocessadores.

Um sistema SCADA é então composto pelos 3 seguintes elementos: Interface Homem-Máquina (HMI); Unidade Central (MTU); e Unidade Remota (RTU).

Interface Homem-Máquina

A interface homem-máquina possibilita, tal como o nome indica, é uma interface gráfica entre o operador e o sistema controlado e supervisionado. Apresenta um sinóptico do sistema, isto é, oferece uma “imagem” do sistema de forma gráfica, a partir de esquemas e/ou figuras

que permitem uma visualização, de forma simultânea, dos seus constituintes e do seu estado de funcionamento. Um exemplo de interface deste género pode ser visualizado na Figura 2.8.

Normalmente, uma HMI permite, para além da visualização do sinóptico do sistema, também um controlo das suas variáveis.

Uma característica importante no desenvolvimento de uma interface deste tipo é o tratamento dos alarmes, pois a detecção de situações anómalas, permitindo que sejam resolvidas rapidamente, é uma das principais funcionalidades de um sistema SCADA. Por este motivo, quando ocorre uma "situação de alarme" no sistema, esta deve ser demonstrada na interface, o mais rapidamente possível, e numa zona de fácil visualização.

Existem ainda, nestas interfaces, a possibilidade de analisar o comportamento do sistema ao longo do tempo, através de um registo histórico e de dados estatísticos, apresentados em valores numéricos ou gráficos, que podem ser utilizados para melhorar o desempenho do sistema, tornando assim os SCADA uma ferramenta bastante atraente também para sistemas de gestão de produção e da qualidade.

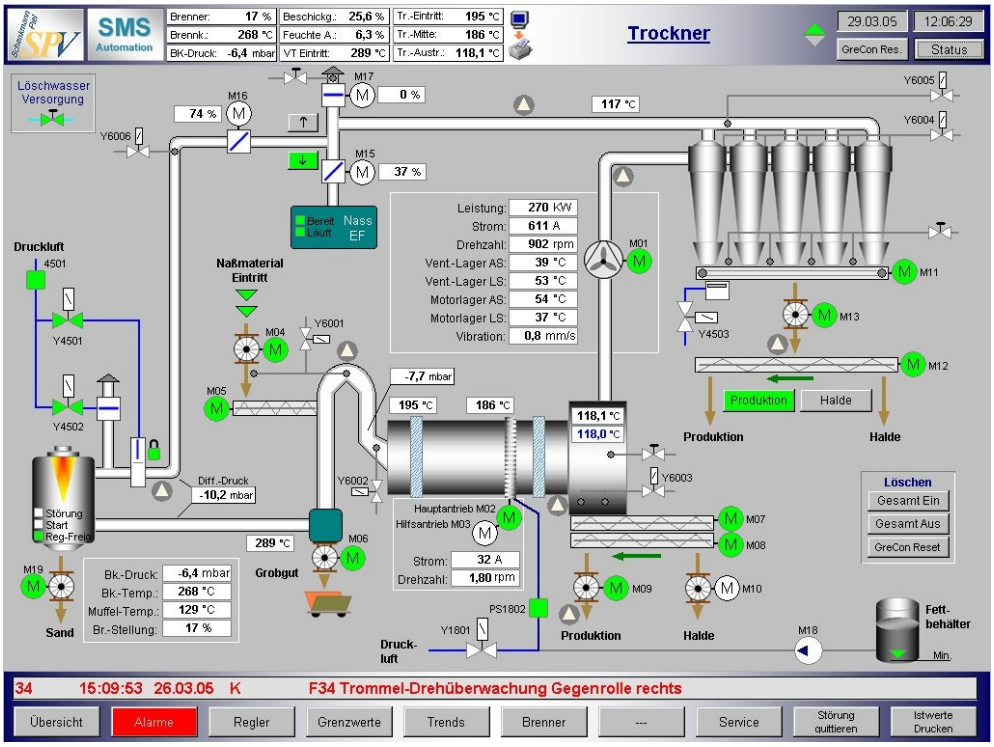


Figura 2.8 – Exemplo de Interface Homem-Máquina desenvolvida no software *SIMATIC WinCC flexible* da *Siemens*

Remote Terminal Unit

Os RTUs são as unidades constituintes de um SCADA situadas mais proximamente do processo que se pretendo supervisionar e controlar. Podem ser utilizado para sistemas simples apenas um RTU, aumentando esse número com a complexidade do sistema, podendo atingir até as centenas de RTUs nos sistemas mais complexos. São utilizadas tanto para recolher os dados dos sensores de campo e envia-los para a unidade central, como para executar, a partir dos actuadores, os comandos recebidos da unidade central. Estas unidades devem ainda conseguir memorizar todas as suas variáveis, permitindo à unidade central aceder-lhes quando quiser. A qualidade de um RTU avalia-se com a fiabilidade, qualidade e segurança dos

dados que transmite. É também, nestas estações que são implementados os algoritmos de controlo básicos locais, ou seja, os algoritmos de controlo activo ou controlo regulatório.

Todas as tarefas pelas quais os RTUs são responsáveis, estão actualmente implementadas nos PLCs, pelo que se utilizam estes processadores em grande parte dos sistemas deste género. No entanto, em vez de PLCs podem utilizar-se também microprocessadores.

Master Terminal Unit

O MTU é a unidade central do sistema. É responsável pela gestão da interface dos HMIs e dos RTUs, funcionando muitas vezes como uma *gateway* de comunicações. Realiza a leitura dos dados recolhidos nas RTUs, traduz esses dados para o contexto do sistema e fornece-os à interface HMI, numa semântica que lhe seja perceptível. Executa também a comunicação em sentido inverso, lendo e interpretando os comandos enviados pela HMI, que posteriormente comunica ao RTU, para que este active os actuadores desejados. Uma outra importante funcionalidade destas unidades é a sua capacidade de guardar, em bases de dados (sistemas de armazenamento), as informações que vão sendo recolhidas ao longo do tempo. Só assim, essas informações podem ser utilizadas pelas HMIs para que estas apresentem, ao utilizador, a evolução histórica e estatística do sistema.

Capítulo 3

Requisitos e Arquitectura do Sistema

3.1 - Requisitos do Sistema

Nesta secção serão listados os requisitos a cumprir neste projecto, assim como os seus objectivos, âmbito, funcionalidades e características. Estes requisitos foram discutidos e definidos pelo autor da dissertação, pelo orientador e pelo colega de projecto, João Ferreira, responsável pela parte de instrumentação do sistema em questão.

Não é objectivo deste projecto a elaboração de um sistema final, que se possa implementar numa estrutura específica, ambiciona-se sim o estudo e validação de várias tecnologias passíveis de serem utilizadas nestes sistemas, que se apresentem como alternativas atractivas às ferramentas, até então utilizadas. Pretende-se assim, o protótipo de um sistema que permita fazer o mesmo, ou se possível mais e melhor que sistemas já existentes, como os da CompactRio ou os da National Instruments referidos na secção 2.1.5 de Sistemas de Aquisição, por menos, ou seja, com menos custos.

Por seguinte, as estruturas que mais preocupam os projectistas, nomeadamente os investigadores de estruturas da FEUP, e que estes mais sentem necessidade de monitorar são as pontes. Por este facto, o trabalho aqui apresentado será direccionado para satisfazer os requisitos de monitorização deste tipo específico de estrutura.

Estes requisitos podem ser divididos, tanto pelos elementos que devem fazer parte do sistema de supervisão e controlo, como pelas tarefas que devem realizar da seguinte forma: sensores e respectivos circuitos de aquisição; actuadores; tarefa de monitorização e controlo do sistema; comunicação entre os vários elementos do sistema; armazenamento dos dados mensurados; e interface gráfica com o operador. Sendo cada um destes requisitos pormenorizado em seguida.

Sensores

Os únicos sensores utilizados neste projecto para validação do seu conceito serão extensómetros, acelerómetros e sensores de temperatura. A sua escolha não foi feita ao acaso, mas sim porque, estes permitem medir algumas das grandezas, destas estruturas, de maior importância para os engenheiros civis, e ainda porque os dois primeiros são em quantidade os mais utilizados numa estrutura supervisionada.

Assim, os requisitos relativos aos sensores passam pelas seguintes tarefas.

- Escolha dos sensores a utilizar. Dependendo desta escolha, os sinais a medir poderão ser de tensão ou corrente;
- Projecto e desenvolvimento dos circuitos de aquisição específicos para cada sensor analógico. Circuitos estes, que devem incluir todo o condicionamento de sinal, bem como, a sua amplificação para adaptação ao conversor analógico para digital utilizado;
- Escolhas dos Conversores A/D a utilizar;
- Estudo do modelo matemático a utilizar nos sinais lidos nos sensores para obtenção das grandezas nas unidades reais, ou seja, encontrar a fórmula de cálculo directo que nos permita determinar o valor da temperatura, em graus *Célsius*, da aceleração em *g* e da extensão em *strains* (deformação), a partir dos valores medidos, em *Volts* ou *Amperes*.

Actuadores

O conceito de estrutura inteligente, apresentado na secção 2.1.3, que se aplica a uma estrutura com actuadores que lhe intervêm de forma controla, em função de valores que avaliam pelos sensores que lhe são também parte integrante, não é ainda muito utilizado em casos reais. Essencialmente no que se refere a actuar sobre a estrutura. Este é um conceito ainda em fase de estudos por parte quer de engenheiros civis, engenheiros electrotécnicos e engenheiros dos materiais. Não é por isso, muito comum actuar directamente sobre grandezas da ponte, sendo que neste caso prático, de sistema de supervisão e controlo, a própria estrutura, representa o seu processo de automação.

Os actuadores possíveis para controlo da ponte podem ser: actuadores hidráulicos de deslocamentos; molas amortecedoras utilizadas em pontes suspensas, que podem ser activadas quando se verifica, pelas medições dos acelerómetros, que os cabos de suspensão estão com elevada ressonância, pois existem casos em que estes cabos podem demorar horas a estabilizar se não tiverem qualquer tipo de ajuda; ou por fim, semáforos de trânsito e/ou cancelas que interditem a circulação na ponte e ainda, no caso dos primeiros, avisem para uma circulação com precaução, ligando o sinal amarelo.

Contudo, neste trabalho não se pretende actuar directamente sobre nenhum parâmetro específico da ponte. Pretende-se sim, que este esteja preparado para o fazer futuramente, reservando para este efeito algumas saídas dos dispositivos do sistema, onde possam ser colocados sinalizadores, como leds, que demonstrem o seu funcionamento.

Os actuadores mais importantes para sistemas deste género são os actuadores de informação, como alarmes, que serão abordados posteriormente.

Monitorização e Controlo

Não se espera que num sistema deste tipo esteja alguém a verificar o seu estado 24 horas por dia, porque durante o projecto de uma estrutura, um dos grandes objectivos dos seus elaboradores é que estas atinjam uma elevada durabilidade e para isso, seguem um conjunto de cuidados que permitam essa característica à estrutura, por forma a que não seja necessária a verificação constante do estado da mesma. Também muitas vezes, essencialmente em obras de grande dimensão, o número de sensores instalados é muito elevado o que torna a sua monitorização em tempo real muito complicado, quer pela quantidade de dados que é necessário transmitir constantemente, quer pelos processadores

que não são tão poderosos ao ponto de conseguirem adquirir e processar todos esses dados constantemente. Mas, o envelhecimento, o desgaste ou a danificação autónoma dos materiais, bem como, a ocorrência de acidentes graves para a "saúde" da estrutura, como o embate de um veículo num pilar de suporte, não podem ser evitados pelos responsáveis da obra. É nestes casos que os sistemas de monitorização podem ser muito úteis, avisando atempadamente as pessoas indicadas de modo a prevenir a ocorrência de situações de dano maior.

O objectivo principal na monitorização de uma estrutura, é então, o aviso atempado de modo preventivo de alguma ocorrência inesperada, que permita aos gestores da estrutura tomar as atitudes que acharem convenientes em cada situação. Este aviso, deve ser realizado por algum meio de comunicação eficaz, como uma mensagem escrita para o telemóvel, ou uma mensagem de email.

É no entanto também, objectivo destes subsistemas permitir a visualização pontual do estado de todas as variáveis monitorizadas, quando solicitado por um operador.

Assim, um sistema deste tipo deve possuir dispositivos inteligentes, que podem ser designados de dispositivos de monitorização e controlo, que permitam a aquisição e processamento dos dados monitorizados; envio de forma autónoma de informação de aviso ao verificar-se a ocorrência de uma situação anómala; envio de dados medidos pelos sensores quando estes lhe forem requeridos e mudar o estado dos actuadores de processo autonomamente ao receberem uma ordem para tal.

Nestes dispositivos, deve então ser aplicado o modelo matemático já mencionado, que permite obter a partir dos dados de entrada dos sensores, o seu valor nas respectivas unidades de medida (ºC, g, ou *Strain*). Estes dados de entrada são respectivos aos dados provenientes dos sensores, mas já em formato correcto (digital), graças aos circuitos de aquisição e particularmente aos conversores analógico para digital.

Quando se obtém o valor nas unidades certas, os dispositivos devem verificar se este corresponde a um valor de alarme do sistema, e em caso afirmativo estes dispositivos devem enviar essa informação o mais rapidamente possível para os responsáveis da estrutura.

Estes mesmos dispositivos, do subsistema de monitorização e controlo, quando estão a realizar medições continuamente pelos seus sensores, devem permitir o envio dessa informação obtida para outro subsistema, a pedido de um operador, sem que isto prejudique o envio atempado de alarmes que pode ocorrer a qualquer instante. Para isso estes dispositivos devem conseguir receber mensagens de pedidos de leitura. Pedidos estes, que devem conter assinalados quais os sensores de que se pretendem receber informação e durante quanto tempo devem efectuar essas leituras. Para tal, deverão ser oferecidas duas formas possíveis para a leitura das grandezas dos sensores ao subsistema interessado: a forma contínua, em que as medições serão realizadas e enviadas para o operador até que este dê ordem para o fim dessa tarefa; ou ainda a possibilidade de medir essas grandezas durante um determinado intervalo de tempo. Um cuidado a ter em conta com estas leituras, tem a ver com o número de sensores que o operador pretende ler, assim o sistema deve estabelecer um número máximo de sensores em que o utilizador pode acompanhar a monitorização, caso contrário a rede poderia ficar "entupida" com estas trocas de informação, originando atrasos significativos no envio de uma mensagem de alarme, que possa ter ocorrido durante este processo de leitura do operador.

É ainda de ter em conta que para a possibilidade de enviar os dados das medições efectuadas durante um intervalo de tempo, será necessário adicionar uma noção de tempo a

estes dispositivos. Para isso pode ser, por exemplo, utilizado um *Real Time Clock* (RTC), que permite aos dispositivos electrónicos manter um controlo preciso do tempo. O RTC tem de ser sincronizado no tempo com todos os elementos do sistema que possuam também um relógio interno, para tal este tem de permitir a recepção e envio de mensagens específicas para essa sincronização.

Uma outra funcionalidade, acima indicada e que estes dispositivos devem oferecer, é a capacidade de alterar o estado de um actuador quando receber um pedido para esse efeito. Para isto, o dispositivo deve também ser configurado para permitir a recepção destes pedidos, onde deve conter bem assinalados quais os actuadores que se pretende alterar e para que estado se pretende alterar. Para existir coerência nesta funcionalidade do sistema, estes dispositivos devem responder de volta à entidade que realizou o pedido, através da confirmação da execução desse pedido, se este for concluído com sucesso; ou então com uma mensagem de erro indicativa da não realização dessa tarefa. Adicionalmente, deve existir a possibilidade de um operador enviar um pedido de leitura do estado em que se encontra um actuador, sem querer modifica-lo, ao qual o dispositivo deve responder correctamente com o estado do actuador pretendido.

O processo de troca de mensagens entre dispositivos do mesmo subsistema, ou entre dispositivos de subsistemas diferentes, pode ser visto como um processo envio e recepção de actuadores de informação. Um caso particular desses actuadores são os alarmes, em que o dispositivo de monitorização e controlo ao verificar a ocorrência de um alarme, deve enviar autonomamente essa informação a dispositivos de subsistemas superiores.

O sistema a desenvolver deve ser projectado para que se utilize no mínimo um dispositivo de supervisão e controlo por pilar, numa ponte com pelo menos 10 pilares, ou seja, com um número de dispositivos maior ou igual a 10.

Comunicação

Uma das principais bases para este sistema de supervisão e controlo é a comunicação de dados entre o sistema de aquisição central, que permite o processamento, e a visualização desses dados ao operador, com os dispositivos de monitorização e controlo (nós) que realizam as medições. Esta comunicação deve ser realizada recorrendo a uma rede de comunicação industrial que tem entre outras: restrições de distância; restrições temporais, relativas à velocidade de transmissão e à latência desde os dispositivos de amostragem até ao centro de processamento; agressões externas; e custo.

Um dos requisitos básicos de uma rede de comunicação para aplicações deste tipo é a distância entre os nós, por exemplo a ponte Vasco da Gama sobre o rio Tejo, a maior ponte em Portugal, tem um comprimento total de 17,3 km. Deste modo, para a realização deste projecto deve ser considero o cenário deste sistema ser aplicado numa ponte com estas dimensões. Não se espera que a rede utilizada permita uma comunicação directa entre as duas extremidades da ponte, podem ser incluídas repetidores na linha, desde que o número destes componentes não seja tão elevado que aumente significativamente o custo de implementação e instalação da rede e que não introduza latência na rede de forma exagerada, ao ponto de por em causa o seu bom funcionamento. A função destes repetidores não é mais que a amplificação e regeneração dos sinais transmitidos na rede.

Como já foi referido, deve ser utilizado pelo menos um nó por pilar, uma boa forma para isto seria utilizar um número de repetidores menor ou igual ao número de nós existentes. Considerando então, como pior caso, o comprimento máximo entre pilares da ponte Vasco da

Gama, com cerca de 420 m, a rede utilizada deve oferecer ligações sem repetidores com esse mínimo de distância.

Um outro requisito, diz respeito à largura de banda/velocidade da rede, ou seja, à quantidade de bits por segundo que a rede consegue transmitir. Este requisito pode ser calculado a partir da quantidade de informação produzida pelos sensores, bem como, pelo período/frequência de amostragem a que se pretendem medir. Considerando então, que aproximadamente, cada sensor produz 4 bytes de informação e assumindo que a ponte terá 10 pilares e que cada pilar contém 100 sensores, no pior dos casos será necessário transmitir para o processador central 4 kbytes de dados num ciclo de amostragem. A frequência de amostragem para esta aplicação não tem necessariamente que ser uma frequência muito elevada, pode se assumir uma frequência de 1 Hz, o que implica períodos de aquisição de 1 segundo. Então no pior cenário, será necessário transmitir 4 kbytes em cada 1 segundo, ou seja, 4000 x 8 bits = 32000 bits, então para a transmissão dos dados relativos aos sensores, é necessária pelo menos, uma largura de banda de 32 kbps. No entanto, normalmente numa mensagem/pacote de dados com informação dos sensores, nunca é incluído apenas o valor obtido nos sensores, mas são regularmente sempre incluídos dados relativos à identificação da origem dos dados, identificação da mensagem, dados de controlo (CRC), entre outros, o que dependem do protocolo utilizado. Assume-se então, que o número de bytes que é necessário transmitir por sensor, não é de 4, mas sim de 10 bytes, a rede passa a necessitar de uma largura de banda de pelo menos 80 kbps.

Nunca serão no sistema real transmitidos para o processador central 100 kbits de dados por segundo, porque é difícil para ele fazer todo o processamento relativo a esses dados num curto espaço de tempo, no entanto, a rede de comunicação deve ser projectada para esse efeito.

A latência (atraso da comunicação) desde a amostragem dos dados até ao centro de processamento é outro aspecto a considerar no projecto de uma rede de comunicação. Este aspecto está directamente relacionado com a velocidade de transmissão de dados na rede e aumenta significativamente, como já foi dito anteriormente com o número de repetidores utilizados, bem como, com a distância a que os dados são enviados de uma para a outra. Para prevenir este aspecto a largura de banda mínima para esta rede de comunicação deve ser aumentada. Introduz-se assim, à velocidade mínima da rede para resolver este aspecto mais 10 kbps, passando deste modo, para um valor de banda larga mínimo de 90 kbps.

O requisito seguinte, diz respeito às agressões electromagnéticas e condições ambientais do meio envolvente. Para prevenir perdas de comunicação por estes últimos factores, estas redes devem estar bem protegidas de qualquer agente atmosférico, como chuvas, ventos, neve, temperaturas baixas ou elevadas, etc. Pode ainda, ser necessária, dependendo da escolha da rede, uma protecção externa ao cabo. Relativamente às agressões electromagnéticas estas estão directamente relacionadas com o meio envolvente, por exemplo, a ponte pode situar-se perto de um aeroporto (dos seus radares); de instalações fabris que incluam muita maquinaria eléctrica; ou de passagens de veículos como o Metro, onde estas agressões são mais significativas. Então, como uma ponte pode situar-se em qualquer um destes locais, a rede que se utilizar deve aguentar estes “ataques” externos.

Finalmente, aquele que é o requisito mais utilizado como critério de desempate é: o custo. Deve, por razões óbvias, escolher-se a solução que satisfaça todos os requisitos anteriormente mencionados e que apresente o menor custo.

Armazenamento dos Dados

É objecto essencial de um sistema de supervisão e controlo uma base de dados, com toda a informação relevante guardada num local em que possa ser consultada a qualquer instante. No caso prático de um sistema de monitorização de estruturas com este histórico de dados, pode ser facilmente verificada a diferenciação ou não, com o tempo das grandezas medidas, permitindo assim visualizar por exemplo, a deterioração da ponte ao longo do tempo.

Pretende-se por isso neste projecto a gravação numa base de dados de todos os alarmes ocorridos; de todas as grandezas que tenham sido pedidas para ler pelo operador aos dispositivos de monitorização e controlo, não necessariamente de todas os dados monitorizados; e ainda de todas as mudanças de estado dos actuadores do processo.

A partir desta base de dados, deve ser produzido periodicamente um relatório com informação relevante ao estado da ponte onde conste: os alarmes ocorridos; as grandezas monitorizadas; e as mudanças de estado dos actuadores que surgiram de novo desde o final do dia anterior.

Interface com o Operador

A interface com o operador (HMI), que será explorada mais há frente, é uma ferramenta gráfica com características específicas, que permite ao operador interagir com o processo de um sistema. As principais características de uma interface com o processo, depreende-se com o facto de permitir ao operador visualizar os dados do processo que pretender e de possibilitar-lhe actuar no processo remotamente.

Uma interface para um sistema como este tem restrições específicas. Deve em primeiro lugar permitir, com ênfase, perceber o estado geral da ponte, isto é, verificar se esta se apresenta estável ou instável. Em segundo, deve apresentar, também de forma realçada os alarmes que possam ter sucedido, numa lista que deve incluir a data e a hora de ocorrência do alarme e ainda uma pequena descrição desse alarme, que possibilite ao operador compreender o que sucedeu. Para além destas duas finalidades, deve conter um sinóptico gráfico da ponte, que permita de forma rápida verificar, para os diferentes locais/pilares em que estão instalados os dispositivos de monitorização e controlo, se esse local se encontra estável ou não.

A partir desta interface o operador deve ainda conseguir fazer pedidos de leitura dos dados monitorizados aos dispositivos específicos, podendo seleccionar os sensores que pretende ler, bem como, o intervalo de tempo em que o deseja fazer. A apresentação dos dados deve ser feita essencialmente de forma numérica, nas unidades respectivas a cada grandeza, mas também, para o caso dos acelerómetros, de forma gráfica.

Por fim, a interface deve permitir ao utilizador ler o estado dos actuadores do processo, bem como, modifica-lo, isto para que lhe seja possível seleccionar o estado que pretende para o actuador, de entre uma lista de possibilidades, de modo a evitar que sejam seleccionados estados inexistentes.

Adicionalmente, deve ainda ser desenvolvido um *Web Site*, com o mesmo formato que a interface, mas não necessariamente tão robusto. Este deve apresentar: o estado geral da ponte; os alarmes ocorridos; o local de ocorrência desses alarmes; os valores monitorizados numa forma numérica, não necessariamente numa forma gráfica; bem como, o estado dos actuadores do processo, podendo ainda permitir a alteração desse estado. Não se pretende com esta interface remota realizar as mesmas tarefas que a interface principal, esta servirá

essencialmente para os responsáveis pela obra monitorizada poderem visualizar o estado global da mesma.

3.2 - Arquitectura do Sistema

3.2.1 - Arquitectura Proposta para o Sistema

No sentido de satisfazer todos os requisitos mencionados em cima, foi proposta, pelo autor e pelo colega de projecto e aprovada pelo orientador, a arquitectura para o sistema, que se encontra representada na figura 3.1.

O sistema é baseado numa arquitectura com processamento distribuído hierarquicamente, por três unidades: a Unidade de Interface, Comando e Controlo, de mais alto nível da hierarquia; Unidade de Comunicação e Controlo Inteligente, o nível intermédio e pela Unidade de Processo, que representará o nível mais baixo do sistema. Na primeira Unidade, realizar-se-ão as tarefas de interface gráfica com o operador, servidor para acesso remoto (por Internet) e ainda o armazenamento dos dados. A Unidade Comunicação e Controlo Inteligente será responsável pela comunicação entre os vários módulos do sistema e também, por tarefas de monitorização e controlo, essencialmente relativas aos alarmes. Por fim, da Unidade de Processo farão parte os actuadores, os sensores e os seus circuitos de aquisição e serão efectuadas ainda, complementarmente à Unidade anterior, tarefas de monitorização e controlo.

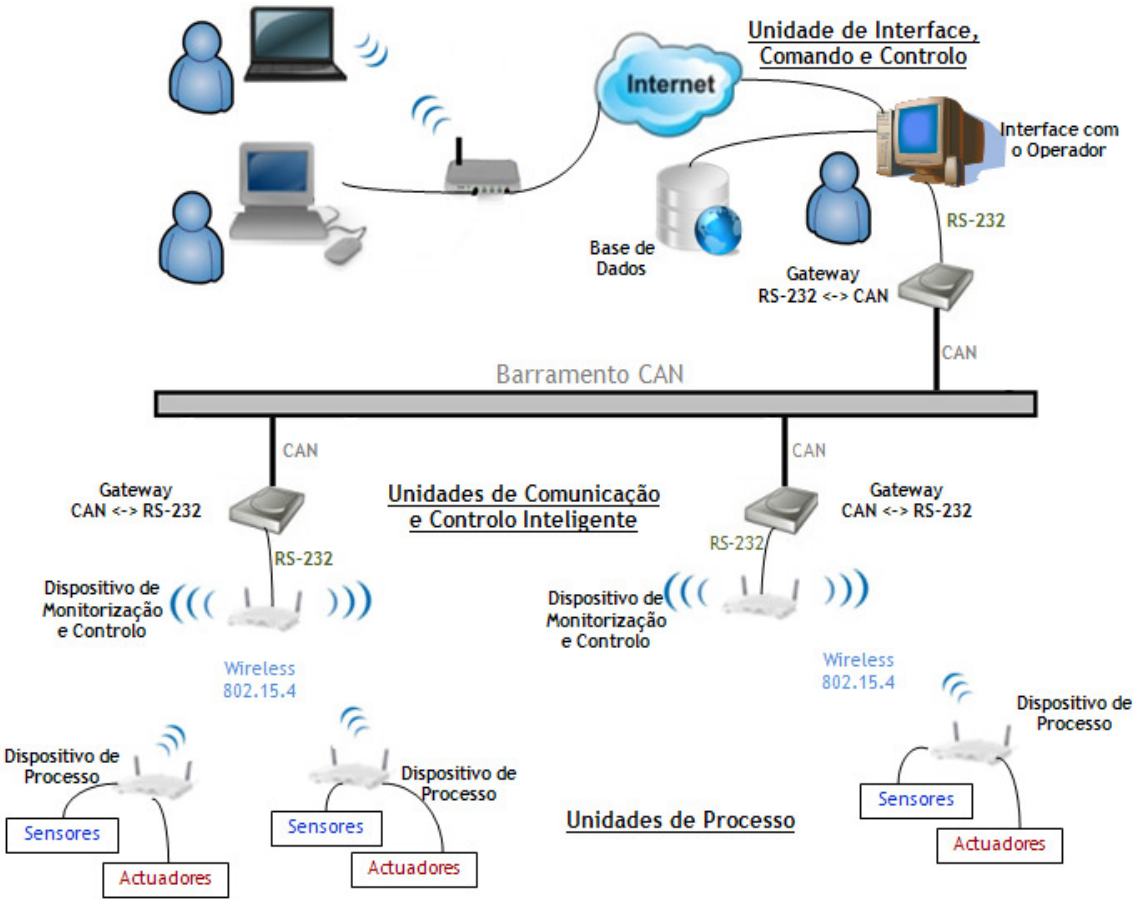


Figura 3.1 - Arquitectura Proposta para o Sistema

Unidade de Interface, Comando e Controlo

Na Unidade de Interface, Comando e Controlo, existirá um computador, conectado à Internet, que será o processador central do sistema de supervisão e controlo. Neste computador será implementada a Interface gráfica (HMI) para o operador, que lhe permitirá realizar todas as tarefas enumerados nos requisitos. Esta interface demonstrará assim: de forma clara e bem visível o estado global da ponte; um sinóptico da mesma onde se poderão verificar de forma gráfica e bem assinalada os locais da ponte que se encontrem estáveis ou não; a lista dos alarmes que possam ter surgido, com um número identificador de cada um, a sua data e a hora de ocorrência e uma pequena descrição do alarme em questão; poder-se-á aceder directamente a cada um dos dispositivos de monitorização e respectivos dispositivos de processo para se visualizar de forma numérica e de forma gráfica (para os acelerómetros) os valores das varáveis monitorizadas; permitirá ler e alterar o estado dos actuadores do processo, que serão apenas dois semáforos; possibilitará ao operador sincronizar os relógios (RTCs), de todos elementos constituintes, com o seu e permitirá ainda abrir e fechar a comunicação entre si e os dispositivos de níveis inferiores. Todos os dados que a interface recebe, sejam de alarmes, resultados das medições, ou estados de actuadores, são introduzidos numa base de dados remota, por comandos SQL (*Structured Query Language*).

Ainda no computador desta Unidade será implementado o servidor Web, que comunicará internamente com a interface HMI, permitindo assim, que algum utilizador autorizado, em qualquer parte do planeta pela Internet, consigo acompanhar a monitorização realizada na interface e ainda alterar o estado dos actuadores do processo. A interface HMI e o Web Site (Cliente e Servidor) serão desenvolvidos recorrendo às linguagens de programação da *framework* .Net da *Microsoft*: C# e ASP.NET, respectivamente.

Unidade de Comunicação e Controlo Inteligente

As tarefas das Unidades de Comunicação e Controlo Inteligentes serão realizadas em dois componentes: as *gateways* CAN – RS-232 e os dispositivos de monitorização e controlo. O primeiro componente, servirá exclusivamente para tarefas de comunicação entre dispositivos, já o segundo para além de tarefas de comunicação, realizará também acções de monitorização e controlo.

Todavia, num sistema de automação industrial distribuído, utilizam-se normalmente três tipos de redes: redes de fábrica; redes de célula e redes de campo. Contudo, no sistema proposto para este projecto serão utilizadas apenas duas: uma rede de controlo, que desempenhará funções da rede de fábrica e da rede de célula e uma rede de campo, idêntica à do sistema de automação distribuído.

Na rede de controlo serão utilizadas duas tecnologias de redes de comunicação distintas: a rede CAN-Bus e a norma eléctrica de comunicação RS-232. A rede CAN-Bus é uma rede que permite satisfazer todos os requisitos de uma rede de comunicação, que já foram em cima apresentados

No entanto, não é muito comum encontrar computadores para a Unidade de Interface, Comando e Controlo, que possuam um interface CAN que lhes permita receber e enviar dados, existem sim para este efeito, adaptadores de USB – CAN, como por exemplo o *Kvaser* 00241-8, ou adaptadores RS-232 – CAN, como por exemplo o *gridconnect CAN-uVCCM*, mas são relativamente caros, por isso foi proposto o desenvolvimento de uma *gateway* de baixo custo,

que permita a conversão das mensagens específicas deste sistema, de acordo com um protocolo estabelecido e dos sinais de comunicação CAN para RS-232 e vice-versa.

Além disso, numa rede CAN, podem ser utilizados cabos eléctricos com comprimento máximo de 1 km, por isso, com um único cabo pode-se atingir o comprimento mínimo requisitado, no então, não é possível com um só cabo cobrir toda a ponte. Pode no entanto configurar-se estas *gateways*, para funcionarem como repetidores da rede, não havendo assim necessidade de utilizar mais que um dispositivo para esse efeito. Para desenvolver estas *gateways*, será utilizado o microprocessador AT90CAN64.

Os microprocessadores utilizados como dispositivos de monitorização e controlo, os *Libelium Waspmotes*, também não possuem uma interface CAN que lhes permita comunicar com o computador da Unidade superior. Mas, contêm entradas UART, que utilizam a mesma norma de comunicação que o RS-232, à excepção da amplitude dos sinais eléctricos, por este facto será utilizada uma *gateway* muito idêntica à anterior, que permitirá dotar, também estes dispositivos de uma interface para uma rede CAN-Bus.

Como rede de campo, será utilizada uma rede ZigBee 802.15.4, que permitirá a comunicação entre os dispositivos de comunicação e controlo e os dispositivos de processo/campo. Esta comunicação, será ao contrário da anterior realizada de forma directa, sem necessidade de qualquer "ponte" de comunicação, isto porque os dispositivos utilizados para um caso e para o outro serão os *Libelium Waspmote*, que possuem uma placa de desenvolvimento já preparada para o emprego dos módulos ZigBee na mesma.

Outros elementos das Unidades de Comunicação e Controlo Inteligente são os dispositivos de monitorização e controlo. Dos requisitos apresentados para estes dispositivos, parte serão realizados nestes elementos e outros nos dispositivos de processo apresentados mais à frente.

Estes dispositivos funcionaram de três formas distintas: como "escravos" do computador da Unidade de Interface, Comando e Controlo; como "mestres" dos dispositivos da Unidade de Processo e como "pontes" de comunicação entre a Unidade superior e a Unidade de Processo.

Para melhor compreender o seu funcionamento, deve-se recordar o modelo de cooperação Mestre – Escravo, utilizado em sistemas distribuídos para cooperação entre os seus equipamentos. Neste modelo os dispositivos mestres tomam sempre iniciativa de iniciar uma comunicação, as mensagens que enviam são normalmente ordens aos escravos para execução de alguma acção, ou pedidos de leitura de alguma variável dos mesmos. Estes escravos, apenas respondem a essas ordens ou pedidos, com mensagens de confirmação ou não da execução das acções pretendidas, ou com mensagens que incluam o valor das variáveis que lhe foram solicitadas.

Assim, os dispositivos de monitorização e controlo, quando funcionam com escravos do computador central, poderão receber mensagens de pergunta, destes mestres, a uma possível ocorrência de alarmes nesse dispositivo ou nos dispositivos de processo que lhe estão associados. A este último pedido, os dispositivos de monitorização e controlo (escravos), se verificaram a ocorrência de alarmes, responderão com uma mensagem que incluía esses alarmes de forma perceptível ao mestre. Ou se não se verifica qualquer alarme, a sua mensagem de resposta deve indicar a não ocorrência de qualquer alarme. Para que o atraso, desde o surgimento de alarme, até ao seu envio ao computador central, seja o mais curto possível, a frequência com que a pergunta de alarme é realizada pelo mestre deve ser elevada, de acordo com o tempo real deste sistema, ou seja, deverá um valor à volta de 1Hz.

Outro modo de funcionamento destes dispositivos, será como mestre autónomo dos dispositivos de processo. Mestre autónomo, pois terá a responsabilidade de questionar os seus escravos (dispositivos de processo a si associados), com um frequência maior que a frequência com que é questionado pelo mestre central, para a possível ocorrência de alarmes nesses elementos. Caso os seus escravos lhe respondam com alarmes, ele deve guardá-los-á na sua memória e enviá-los-á à Unidade superior no próximo pedido para leitura de alarmes que esta lhe efectuar.

Por fim, estes dispositivos funcionarão ainda, como "pontes" de comunicação entre o computador central e os dispositivos de processo. Este fenómeno ocorrerá, quando o computador central pretender ler algum dos sensores, ou mudar o estado de um actuador dos dispositivos de processo, que estão associados aos dispositivos de monitorização e controlo. De tal forma, estes dispositivos ao receberem um pedido, da Unidade de Interface, Comando e Controlo, para por exemplo, uma leitura do valor medido por um sensor relativo a um dos dispositivos de processo a este associado, reencaminharão essa mensagem para o dispositivo pretendido, que lhe responderá com o valor pretendido, para que este o reenvie à origem do pedido, isto é, ao computador central do sistema.

Uma situação em que os dispositivos de monitorização e controlo poderão funcionar das três formas distintas, como escravos, mestres ou "pontes", será quando receberem, do computador de comando do sistema central, ordens para sincronização do seu e dos relógios dos dispositivos de processo a si associados. Nesta situação, estes dispositivos deverão sincronizar o seu relógio interno e em seguida enviar uma mensagem de sincronização dos RTCs a todos os dispositivos de processo a si associados. Se receber de todos esses dispositivos respostas de confirmação da execução desta tarefa e se ele mesmo a conseguir executar, responderá ao computador central, com uma mensagem de confirmação da sincronização de todos estes relógios.

Num projecto do sistema inicial, tinha sido planeado utilizar os dispositivos de monitorização e controlo também para realizar medições por sensores e actuar na ponte utilizando actuadores, ou seja, para funcionarem também como dispositivos de processo, visto serem utilizados para o seu desenvolvimento microprocessadores especializados para sensorização, como são os *Libelium Waspmotes*. Caso contrário, não faria muito sentido utilizar estes componentes para realizar tarefas, apenas de comunicação e processamento, porque poder-se-iam utilizar processadores mais baratos para esse efeito. No entanto, como o número de sensores era tão limitado, optou-se por utiliza-los apenas como unidades de comunicação e controlo, que comunicam numa rede sem fios com os dispositivos de processo/campo e com o computador central do sistema.

Unidade de Processo

Para implementar estas Unidades, serão utilizados, também os microprocessadores especializados para tarefas de sensorização e comunicação por redes sem fios, *Libelium Waspmotes*.

Estas unidades realizarão as seguintes tarefas dos requisitos de monitorização e controlo: verificação da ocorrência de alarmes pelas medições efectuadas; resposta a pedidos de leitura. dos valores medidos nos seus sensores, por parte de Unidades superiores; execução de ordens recebidas para alterar o estado dos seus actuadores; sincronização do seu relógio a partir de mensagens de sincronismo dos dispositivos do nível superior e ainda, a aplicação do

modelo matemático que permitirá, em função do valor em bits lidos do ADC do circuito de aquisição dos sensores, obter o seu resultado nas unidades de medida respectivas.

Configurando estes dispositivos para realizar as tarefas aqui referidas, são satisfeitos os requisitos de monitorização e controlo, que não fazem parte das Unidades de Controlo Inteligente.

Cada uma destas unidades de processo, ou dispositivos de processo, como podem também ser designados, estará associada a um único dispositivo de monitorização e controlo e comunicarão entre si, utilizando uma rede sem fios ZigBee 802.15.4.

As quantidades e tipos de sensores que estão previstos para serem utilizados, são os seguintes: seis extensómetros; quatro acelerómetros, sendo dois deles os acelerómetros incorporados nas placas de desenvolvimentos dos Waspmotes; e dois sensores de temperatura (RTDs), que serão distribuídos pelos dispositivos de campo existentes. Foram, anteriormente ao inicio da execução do projecto, adquiridos cinco Waspmotes, que deverão todos eles ser aplicados neste sistema. Dois deles serão utilizados como dispositivos de comunicação e controlo inteligente, e os restantes três serão utilizados como dispositivos de processo. A distribuição dos sensores pelos três dispositivos de campo será a seguinte: um deles suportará os seis extensómetros e os outros dois possuíram, cada um, dois acelerómetros e um sensor de temperatura.

Os sensores aqui referidos dizem respeito, aos sensores que foram cedidos para a realização de testes deste projecto, no laboratório de estruturas do departamento de Engenharia Civil (LABEST).

Quanto aos actuadores serão, utilizados apenas em dois destes dispositivos de campo, três leds de cores verde, amarela e vermelha que representaram dois possíveis semáforos da ponte.

Sistema Global

A arquitectura proposta pode parecer demasiado complexa quando se pensa que, o sistema a desenvolver apenas medirá valores de doze sensores e manipulará dois actuadores. No entanto, convém relembrar, que com este projecto se pretende o design de um sistema protótipo, que possa futuramente ser aplicado numa ponte real, onde seja necessária a monitorização de milhares de sensores, bem como, o manuseamento de vários actuadores. E por não ser objectivo do trabalho, não serão adquiridos novos sensores e actuadores, visto o principal propósito para este, projecto o ser a validação de vários conceitos.

Olhando agora, de um modo geral, para a arquitectura proposta (figura 3.1), pode resumir-se muito rapidamente que o sistema, que tem como base o modelo de cooperação Mestre – Escravo, é composto por uma Unidade de processamento central - Unidade de Interface, Comando e Controlo, desenvolvida num computador, que pode ser designado de Mestre Central. A este Mestre Central, estão associadas duas Unidades de Comunicação e Controlo Inteligente, podendo ser denominadas de Mestres e distinguidos como Mestre 1 e Mestre 10, que no fundo são escravos do Mestre Central. E ainda três Escravos, dois (Escravo 1 e Escravo 2) associados ao Mestre 1 e um outro (Escravo 10) associado ao outro Mestre 10, os quais representam os dispositivos de campo ou processo.

É ainda de salientar mais uma vez que, o grande objectivo de um sistema de supervisão e controlo de estruturas, é a sua capacidade de avisar os responsáveis pela obra da ocorrência de alarmes. Porém para isso, é necessário numa fase de design do sistema, definir os

diferentes tipos alarmes que devem ser tratados neste sistema. Sendo assim, os alarmes que serão tratados pelo sistema, são os seguintes:

- Verificação da passagem de um limite superior ou inferior, previamente estabelecido, a uma das grandezas medidas nos sensores dos dispositivos de campo, como por exemplo as vibrações da ponte atingiram os 2g;
- Verificação de que algum dos sensores estará a realizar medidas sem sentido, por estar danificado, por exemplo o sensor de temperatura medir 100ºC de temperatura ambiente;
- Verificação de que a bateria de alimentação de algum dos dispositivos das Unidades de Processo (unidades de sensores sem fios) se encontra com pouca carga, por exemplo com apenas 10% da sua carga máxima possível;
- E por último, quando se verifica o surgimento de uma falha de comunicação, por não se ter obtida qualquer mensagem de resposta a uma dada mensagem enviada. Este alarme será verificado apenas pelo Mestre Central.

É requisito deste sistema que os alarmes sejam verificados e enviados, o mais rapidamente possível, aos responsáveis pela obra. Por isso, se este apenas surgirem na interface, corre-se o risco de não serem visualizados por ninguém, pois como já foi dito, não se espera que exista um operador a monitorizar a interface 24 sobre 24 horas. A solução para este problema será o envio de um e-mail, aos responsáveis pela obra, o mais rapidamente possível após o aparecimento de um alarme.

Serão ainda, produzidos diariamente pela interface, relatórios de informação relativa aos alarmes, às medições efectuados e às mudanças de estado dos diferentes actuadores nesse dia. Esse relatório será depois de produzido, enviado aos responsáveis pela obra também por correio electrónico.

3.2.2 - Escolha das Ferramentas e Componentes do Sistema

Interface Homem – Máquina

Existem muitas ferramentas, com licenças gratuitas, possíveis para o desenvolvimento da interface gráfica, que permitem ao operador interagir com o sistema, das quais podemos destacar: as principais linguagens de programação da *framework* .NET (*dotNet*) da Microsoft, que são C# (CSharp), Visual Basic.Net e Visual C++; a plataforma Lazarus, programável em Free Pascal; a plataforma Qt, programável em C++; a linguagem Java e ferramentas de programação grátis especializadas para o desenvolvimento de HMIs de SCADAs, como o FreeSCADA, programável em C#.

A diferença entre as linguagens de programação da plataforma .NET, da Microsoft, reside inteiramente na sua sintaxe, porque pode obter-se o mesmo produto final utilizando as 3 linguagens distintas. O grande objectivo da Microsoft com o desenvolvimento desta *framework*, era exactamente que todo e qualquer código gerado em .NET, possa ser executado em qualquer dispositivo que possua este mesmo *framework*. Então, quando se opta por desenvolver uma aplicação utilizando esta plataforma, a escolha da linguagem que se utilizará para programar, é inteiramente, de entre estas três, a que o programador preferir. A linguagem Visual Basic.NET é uma linguagem de programação, completamente orientada a objectos e com uma forma de programar muito distinta da antiga Visual Basic 6.0, não podendo por isso afirma-se que um programador especializado em Visual Basic, seja capaz de programar em Visual Basic.NET. Relativamente à linguagem de programação Visual C++, é a

mais recente linguagem desta plataforma, aqui as aplicações podem ser concebidas utilizando a conhecida linguagem C++. Por sua vez, a linguagem de programa C#, que foi desenvolvida com algumas das melhores características de outras linguagens, como Java, Object Pascal e C++, e permitiu ainda, corrigir alguns dos seus principais problemas. Assim, desenvolver uma aplicação em C# é uma tarefa mais fácil do que utilizando, por exemplo a linguagem C++, porque a sintaxe desta linguagem é mais simples [18]. Muitos dos utilizadores da linguagem Java e até de C++, migraram para o .NET por preferirem a linguagem de programação C#.

A grande desvantagem desta plataforma é que apenas pode ser utilizada em sistemas operativos Windows. Para desenvolver aplicações baseadas nesta plataforma, utiliza-se o ambiente de um desenvolvimento especializada da Microsoft, o Visual Studio .NET.

Desenvolver aplicações gráficas utilizando esta plataforma é, comparativamente a outras, bastante simples. Para a interface gráfica que se pretende desenvolver, esta plataforma pode ser muito interessante, pois apresenta praticamente todas as ferramentas gráficas e de programação necessárias, como: caixas de texto; botões; tabulações; caixas de verificação; caixa de escolhas múltiplas; apresentação de tabelas; possibilidade de integração de imagens; possibilidade de desenhar imagens simples, desenho de vários tipos de gráficos; fácil utilização de temporizadores; entre muitas outras. Para além disso, possui uma quantidade enorme de bibliotecas, que permitem a interacção desta plataforma com ferramentas externas, como bases de dados; ficheiros de texto; emails; portas comunicação série; Web Sites; etc.

Lazarus é uma outra plataforma bastante poderosa para o desenvolvimento de aplicações gráficas. Possui também bastantes ferramentas quer gráficas, quer textuais, para o desenvolvimento da aplicação em si, tal como, as linguagens do .NET e possui também várias bibliotecas para o desenvolvimento de aplicações, que podem interagir com componentes externos a esta. Pode ainda, ser utilizada em diversos sistemas operativos como: Linux; Windows; Mac OS; ARM e mais.

A framework QT é uma outra plataforma possível para o desenvolvimento da interface, que possui também, uma enorme variedade de ferramentas para a elaboração de aplicações gráficas e permite a interacção com uma enorme variedade de ferramentas externas, tal como, em .NET. Pode ser utilizada nos sistemas operativos mais comuns (Windows, Linux e Mac OS) e é actualmente, muito utilizada para desenvolver aplicações para telemóveis. Uma desvantagem desta plataforma é o facto de, as suas aplicações terem que ser desenvolvidas em C++, o que torna essa tarefa mais morosa que para outras plataformas.

Uma outra possibilidade é o desenvolvimento da interface na linguagem de programação Java, que é uma das linguagens de programação orientadas a objectos mais utilizada por programadores. Tem também, todas as ferramentas necessárias para a elaboração de uma aplicação deste tipo, e pode ser utilizada também, em praticamente todos os sistemas operativos incluindo Mac, Linux e Windows.

Por sua vez, a ferramenta FreeSCADA, como muitas outras ferramentas grátis deste género, é um pouco mais limitada, nomeadamente para a interligação entra esta e o processo do sistema em si. Sendo normalmente necessário, tal como, neste caso em particular, o desenvolvimento de uma comunicação segundo a norma OPC (*Object Linking and Embedding for Process Control*) para esse intercâmbio de dados. Tarefa esta, que tornaria o desenvolvimento da aplicação bastante mais complexa.

Em suma, de entre todas estas, a linguagem utilizada para o desenvolvimento da aplicação será a linguagem C# da plataforma Microsoft .NET. Isto por ser uma linguagem

muito completa e fácil de utilizar, muito fácil, que possui o melhor de outras linguagens de programação, e que permite uma fácil interacção com ferramentas externas, particularmente com o Web Sites desenvolvidos em ASP.NET, que será a seguir apresentado. Como uma aplicação de HMI deve, obrigatoriamente, funcionar em sistema operativo Windows [16], não necessitando de funcionar noutros sistemas operativos, esta ferramenta satisfaz todos os requisitos necessários à implementação da interface desejada.

Web Site

Para desenvolver o Web Site, que permita o acesso via Internet, ao sistema existe também, um variadíssimo conjunto de ferramentas que podem ser utilizadas como: ASP.NET; Google Web Toolkit e também ferramentas especializadas para interfaces homem-máquina Web como o IntegraXor ou o Control System Works.

O ASP.NET é uma plataforma de desenvolvimento de aplicações Web, criada pela Microsoft, que pode ser desenvolvida nas linguagens C# e Visual Basic.NET da plataforma .NET. Normalmente, utiliza-se o mesmo ambiente de desenvolvimento que nas linguagens da *framework* .NET, o Visual Studio .NET. Com ASP.NET podem também ser utilizadas praticamente as mesmas ferramentas de desenvolvimento gráfico que as das linguagens de programação .NET.

Relativamente ao Google Web Toolkit (GWT), é uma ferramenta que permite o desenvolvimento de aplicações Web em Javascript, o que possibilita uma actualização das aplicações Web em Tempo Real, sem a necessidade de um *refresh* à página, por parte do utilizador. Apresenta também, um conjunto enorme de ferramentas gráficas que podem tornar mais atractivas as páginas Web.

Por seguinte, as ferramentas IntegraXor e Control System Works, são bastante interessantes, ambas permitem ligações simples a base de dados e uma integração simples com o processo. No entanto, é necessário adquirir licenças destes produtos para desenvolver interfaces com alguma complexidade, uma vez que a versão gratuita tem algumas limitações no desenvolvimento de aplicação, bem como, no acesso a alguma documentação relativa ao funcionamento destas mesmas ferramentas.

Contudo, para conceber o Web Site do sistema será utilizado ASP.NET. As suas grandes vantagens em relação ao GWT, residem no facto de permitirem uma comunicação muito simples entre o seu servidor e a interface para a visualização dos dados a apresentar no Web Site e, ao contrário do GWT, não há necessidade de, durante a concepção do Web Site, escrever dois códigos, um relativo ao Servidor onde se encontra a informação e outro relativo ao cliente, que será o browser de acesso a esse servidor. Pode ainda assim, pensar-se que as aplicações ASP.NET têm a desvantagem, em relação aos aplicativos GWT, de não ser em tempo real, isto porque não são escritos em Javascript. Há no entanto, a possibilidade de integrar componentes AJAX (*Asynchronous Javascript and XML*) no código ASP.NET, o que pode disfarçar um pouco esta realidade. Um exemplo disso é a utilização de temporizadores AJAX, que em intervalos de tempo definidos fazem a actualização da página automaticamente.

Rede de Controlo

No sentido de escolher uma rede de controlo para o sistema e tendo em conta os requisitos apresentados na secção 3.1, será realizado um estudo comparativo de quatro tipos de redes. Esse estudo pode ser verificado na tabela 3.1.

Tabela 3.1 — Comparação entre possíveis Redes de Controlo

	Ethernet	**CAN-Bus**	**Fibra Óptica**	**RS-485**
Níveis do Modelo OSI	Físico e MAC	Físico e Ligação (MAC e LLC)	Físico	Físico
Largura de Banda Máx.	10 Mbps	100 kbps*	≈ 40 Gbps	100 kbps*
Comprimento Máx. dos Cabos	100 m	500 m*	Centenas de Kms	1200 m*
Resistência a Agressões Ext.	Baixa	Elevada	Muito Elevado	Elevada
Custo Equip.	Médio	Baixo	Muito Elevado	Muito Baixo

* A largura de banda da rede CAN e da Rede RS-484 dependem do comprimento dos seus cabos - os valores apresentador são os que mais se adaptam aos requisitos pretendidos

Pela análise da tabela 3.1, pode excluir-se de imediato a possível escolha de uma Rede Ethernet, uma vez que, não cumpre os requisitos: de dimensão mínima dos cabos de 420 m e de elevada resistência a agressões externas.

Quanto à fibra óptica, é uma rede bastante utilizada em sistemas de monitorização de estruturas físicas, por ser a solução mais robusta e fiável. No entanto, há a necessidade de encontrar um concorrente mais barato quer no custo do material, quer no custo de implementação, que são elevados para esta rede. Isto porque, ela apenas define a camada física do modelo OSI, o que torna bastante complexa a sua implementação e o controlo do fluxo dos dados na mesma, considerando que a esta rede podem estar conectados dezenas de dispositivos.

Restam assim a rede CAN-Bus e a rede RS-485, que satisfazem todos os requisitos mínimos pretendidos para a rede de comunicação desejada e apresentam contrabalançado o custo do equipamento (mais barato no caso do RS-485) com o custo de implementação (menor para CAN-Bus por possuir a também a camada OSI de ligação). Por ser ainda assim, uma rede barata; garantir controlo do fluxo de dados em si mesma; ser mais simples de implementar; e não ter sido ainda uma solução aplicada em monitorização de estruturas, a opção será a rede CAN-Bus.

Microprocessador CAN

Será necessário conceber, como ficou já evidente na arquitectura proposta para o sistema, umas *gateways* RS-232/UART para CAN e vice-versa, para isso, é necessário recorrer a microprocessadores que suportem as duas tecnologias.

As soluções mais baratas, entre cinco marcas diferentes, são as demonstradas na tabela 3.2, a qual ilustra as características de memória e a quantidade de portas UART e CAN, que cada marca apresenta.

Tabela 3.2 — Comparação entre possíveis microprocessadores com interface UART e CAN

	Microchip PIC18F258	**NEC UPD78F0881GB**	**Atmel AT90CAN64**	**Freescale Semiconductor MC9S12D64CFUE**	**Silicon Laboratories C8051F043-GQ**
Memória do Programa	16 kbytes	32 kbytes	64 kbytes	64 kbytes	64 kbytes
Memória RAM	1,5 kbytes	2 kbytes	4 kbytes	4 kbytes	4 kbytes
CPU	8 bits	8 bits	8 bits	16 bits	8 bits
Nº portas UART	1	1	2	1	2
Nº portas CAN	1	1	1	1	1
Preço	5,64 €	4,12 €	11,10 €	12,49 €	12,90 €

Pela análise da tabela, verifica-se que a Microchip é a de todas, a pior solução, por apresentar microprocessadores com apenas 16 kbytes de memória para desenvolver o programa e só 1,5 kbytes de memória RAM, sendo ainda, mais caros que os microprocessadores da NEC que apresentam RAM e memória do Programa superiores a este.

Olhando agora para os microprocessadores na gama dos 64 kbytes de memória do Programa, pode concluir-se, que de entre estes o AT90CAN64 é a melhor solução. Pois, em primeiro lugar, é mais barato que o microprocessador da Silicon Laboratories, que lhe é em tudo semelhante. E em segundo lugar, apesar de o microprocessador da Freescale Semiconductor ser de 16 bits, o AT90CAN64 é ainda melhor solução, por ser mais barato, visto não se justificar pagar mais para utilizar um CPU dessa natureza num dispositivo que apenas servirá de *gateway*.

Entre o microprocessador da NEC e o microprocessador da Atmel, optou-se pelo da Atmel, isto porque, não é possível prever à partida a quantidade de memória de Programa e RAM que será necessária para a implementação da gateways, e caso a opção de escolha fosse para os microprocessadores da NEC e a sua memória não fosse suficiente, seria necessário adquirir outros processadores de outra marca, visto este ser o microprocessador com a memória máxima que a NEC apresenta. Deste modo, a aquisição de um novo dispositivo de outro fabricante implicaria, a inutilidade do programa até então desenvolvido, porque cada microprocessador, ou mais especificamente cada marca, utiliza registos para programar os seus componentes completamente diferentes e também implicaria obviamente, o desembolso de mais dinheiro para a sua aquisição. Então, não correndo riscos, optou-se pelo AT90CAN64, até porque se mesmo os 64 kBytes não forem suficientes, poder-se-á ainda adquirir o AT90CAN128, com 128 kBytes de memória programável, sendo que aqui, já se poderia utilizar o mesmo código que se estava a utilizar para o processador de 64 kBytes.

Rede de Campo

Como redes de campo, estava definido previamente à realização do projecto a utilização de uma rede *wireless*. E faz todo o sentido estudar a utilização de uma rede sem fios numa aplicação desta natureza, uma vez que, como já foi sendo referido grande parte dos custos associados à implementação de um sistema de monitorização em estruturas está relacionada com os cabos utilizados para as comunicações.

Porém, um dos maiores problemas de uma rede de sensores sem fios, é a alimentação dos seus constituintes. Isto porque em redes cabladas, pode no mesmo cabo estarem incluídos cabos com alimentação para os seus circuitos, mas no caso dos sensores sem fios são necessárias baterias para alimentar os circuitos, sendo que estas baterias terão que ser recarregadas com alguma frequência, o que pode não ser uma tarefa muito fácil, como por exemplo, no caso destes circuitos se encontrarem em locais de difícil acesso. Desta fora, podem existir assim, casos em que a tecnologia de redes sem fios não seja uma boa solução, não só pela necessidade de baterias, o que pode não se justificar em sistema de pequena dimensão, mas também, por exemplo no caso de ambientes com muito ruído electromagnético, ou ainda, onde a transmissão de dados seja necessária para realizar a longas distâncias.

Por tudo isto, neste projecto serão utilizadas duas tecnologias até agora ainda não aplicadas em sistemas de monitorização de estruturas, as redes *wireless* e a rede CAN-Bus. E é de salientar que, o projecto poderia ser realizado todo utilizando apenas uma delas, o que reduziria bastante o seu custo e até simplificaria a sua concepção, no entanto, optou-se por uma solução com as duas redes, pois como já foi dito, o grande objectivo desta tese é o estudo de novas tecnologias possíveis de serem aplicadas em sistema deste género.

Microprocessadores de Sensores Wireless

As unidades de sensores wireless a utilizar neste sistema, também não foram escolhidas pelo autor desta dissertação, pois já tinham sido adquiridas previamente ao inicio do projecto. Estes serão os Waspmotes, da empresa Espanhola Libeluim. No entanto, é interessante fazer um estudo, comparando diversas soluções para estes *Motes*. Estão assim, representados na tabela 3.3 algumas unidades de sensores *wireless* possíveis.

De entre as soluções apresentadas na tabela, os Waspmotes (figura 3.2) são os que apresentam o processador mais simples (8 bits). É ainda, o único que permite, comunicações Wi-Fi (IEEE 802.15.1) e ZigBee (IEEE 802.15.4), assim como, trabalhar na banda dos 900 MHz. Nas comunicações *wireless* deste projecto, será utilizada a norma 802.15.4 com módulos XBee, o que implicará uma largura de banda máxima de 250 kbps. Lamentavelmente, os preços destes equipamentos não estão disponíveis, por isso não são apresentados.

Os Waspmotes são assim, uma boa solução para a implementação dos dispositivos de processo do sistema, por não terem que realizar tarefas muito complexas. Já, como dispositivos de comunicação e controlo inteligente, como serão utilizados neste sistema, deixam algumas reservas. Sendo que, no sistema que se propõem a desenvolver, estes dispositivos “mestres” estarão associados apenas a um ou dois “escravos”, mas numa aplicação mais realista, eles poderiam ser associados a algumas dezenas desses escravos, dependendo da dimensão do sistema. Por esse facto, a sua limitação de processamento, pode não ser suficientemente poderosa para controlar todos estes escravos, seria talvez, mais indicada nesta situação a utilização de um dispositivo mais poderoso como o iMote2.

Por fim, as baterias que estes, e quaisquer outros microprocessadores aqui mostrados, apresentam, não são de todo passíveis de serem utilizadas num ambiente real, pois correr-se-ia o risco de ter que as carregar ou substituir com uma frequência muito grande. No caso dos Waspmotes quase diariamente. Então será necessário, em sistemas reais, a utilização de baterias que permitam armazenar muito mais carga, que aguentem estes dispositivos e os seus circuitos de aquisição durante um período de tempo significativo.

Tabela 3.3 — Comparação entre possíveis microprocessadores de sensores wireless

	Intel iMote	**Crossbow TelosB**	**Moteiv Tmote Sky**	**Crossbow iMote2**	**Libelium Waspmote**
Processador	Zeevo ARM7TDMI	TI MSP430	TI MSP430 F1611	Intel PXA271 XScale	Atmel ATmega1281
Memória do Programa	512 kbytes	48 kbytes	16 kbytes	256 kbytes SRAM	128 kbytes
Memória RAM	64 kbytes	10 kbytes	10 kbytes	32 Mbytes SDRAM	8 kbytes
CPU	32 bits	16 bits	16 bits	32 bits	8 bits
Norma Wireless	IEEE 802.15.1	IEEE 802.15.4	IEEE 802.15.4	IEEE 802.15.4	IEEE 802.15.1/ IEEE 802.15.4
Freq. De Banda	2.4 GHz	2.4 GHz	2.4 GHz	2.4 GHz	2.4 GHz, 900 MHz
Largura de Banda Máx.	600 kbps	250 kbps	250 kbps	250 kbps	Depende da norma
Nº ADCs	-	8	8	-	7
Resolução ADCs	-	12 bits	-	-	10 bits
Fonte de Energia	Baterias	2 x baterias AA	2 x baterias AA	3 x baterias alcalinas AA	Bateria de Lítio (3.3V)

Figura 3.2 - Aspecto do Libelium Waspmote

Módulos XBEE

Os XBEE, são uns módulos de comunicação sem fios, que implementam a norma 802.15.4/ZigBee. São componentes de baixo custo e baixo consumo, especialmente concebidos para redes de sensores sem fios, que operam na banda de frequência dos 2.4 GHz e comunicam ponto a ponto [19]. Com estes módulos é possível, portanto implementar as camadas física e de ligação de dados (LLC e MAC) do modelo OSI e consegue-se comunicar com largura de banda de 250 Kbps.

Os Waspmotes, vêm originalmente preparados para implementar a norma 802.15.4 recorrendo a estes módulos XBee. Contendo no seu API bibliotecas que permitem para além de os configurar, também realizar tarefas de comunicação entre si. Existem dois tipos de XBEE, ambos passíveis de ser implementados nos Waspmotes, o XBEE e o XBEE-PRO, que têm como principais diferenças: o custo, o consumo e a distância máxima a que conseguem comunicar. O XBEE é o mais barato, consegue comunicar a uma distância máxima de 500 m e o consumo. O XBEE-PRO é mais caro, no entanto permite comunicar até 7000 m, apresentando um consumo superior, visto este estar directamente relacionado com a distância a que transmite.

Como todos os módulos de comunicação *wireless* estarão todos relativamente perto (distâncias nunca superiores a 500 m) uns dos outros em "ilhas" de dispositivos de processo que comunicam unicamente com uma Unidade de Controlo, por exemplo uma ilha por pilar e porque nesta aplicação, se deve procurar reduzir os consumos ao máximo, as baterias aguentarem o máximo tempo possível, a opção passou pela utilização dos dispositivos XBEE normais.

3.2.3 - Protocolo de Comunicação Geral

Antes de introduzir os capítulos da implementação do sistema, parece ainda importante a definição do protocolo para a comunicação entre as várias Unidades deste Sistema de Supervisão e Controlo em Instrumentação Distribuída, aplicado a Estruturas de Engenharia Civil.

Este protocolo, contém um conjunto de diferentes tipos de mensagens, que permitem realizar todas as tarefas pretendidas para o sistema a desenvolver. O mesmo, não poderá ser utilizado pelas comunicações CAN, como ficará visível mais à frente. Porém, as mensagens estabelecidas neste protocolo, serão as mensagens que as diferentes Unidades do processo receberão e transmitirão. Podendo assim, assumir-se que este será o protocolo utilizado nas comunicações série (RS-232 e *wireless*).

Importa relembrar, que no sistema existe um Mestre Central, um conjunto de Mestres, que são escravos do Mestre Central e um conjunto de Escravos, que são subdivididos em conjuntos mais pequenos e associados a um Mestre. A iniciativa de efectuar uma comunicação é sempre realizada pelo Mestre Central para um dos Mestres, ou dos Mestres para o sistema, podendo ainda, o Mestre Central comunicar com os Escravos mas, para isso, a mensagem terá sempre de passar pelo Mestre ao qual o Escravo está associado.

O mestre, a cada mensagem que envie inicia um temporizador, que só parará quando a resposta a essa mensagem é recebida ou quando atingir o seu valor limite, normalmente de 5 segundos, mas, que se diferencia em algumas tramas. Este temporizador funciona como *timeout*. Quando ocorre um *timeout,* o Mestre Central voltará a tentar enviar a trama que

não recebeu resposta, repetindo este processo três vezes. Ao quarto *timeout* o Mestre Central activa um alarme de *timeout*.

Existirão então, no sistema sete tipos de tramas distintas:

- Tipo W – para leitura de alarmes que possam ter ocorrido;
- Tipo C – para sincronização dos RTCs dos dispositivos do sistema;
- Tipo F – para definição das funções e activação de sensores de um Escravo;
- Tipo R – para leitura dos valores medidos pelos sensores activos num Escravo;
- Tipo L – para leitura de vários valores consecutivos, medidos num sensor específico de um Escravo.
- Tipo A – para leitura do estado dos actuadores num Escravo específico;
- Tipo M – para mudanças de estado dos actuadores pertencentes a um Escravo.

Todas estas tramas seguem, o cabeçalho e final, padrão apresentado na figura 3.3.

Figura 3.3 – Cabeçalho e final padrão das tramas do Protocolo Geral

As tramas iniciam-se sempre com o caracter '~' e todos os parâmetros nela presentes são separados pelo caracter '#', como se pode verificar na figura em cima.

Seguido do caracter indicador de inicio de trama, aparece o identificador do Mestre a que a trama é destinada, ou de quem é efectuada uma resposta, que pode, para o sistema proposto, ser 1 ou 10.

Posteriormente, encontra-se o identificador do Escravo, que se for aplicável, será de 1, 2 ou 10, e tem a mesma função que o ID do Mestre. Este parâmetro, não é relevante, no caso das tramas enviadas pelo Mestre Central que tem como destinatário apenas os Mestres, como são o caso, das tramas tipo W e tipo C.

O parâmetro que se segue é o identificador da mensagem, este parâmetro será necessário, como ficará evidente em capítulos posteriores, para a Unidade de Comando principal por ter uma noção das mensagens que ainda esperam obter uma resposta. Uma vez que, após enviar um pedido, a Unidade de Comando não ficará bloqueada à espera da resposta, podendo, no intervalo entre envio do pedido e o recebimento da resposta, enviar outra mensagem, é lhe importante saber quais as mensagens que ainda espera receber. Portanto, ao ser enviada, pelo Mestre Central, uma mensagem com um identificador específico, é esperada uma resposta a essa mensagem com o mesmo identificador.

O tipo de trama, que aparece a seguir ao identificador da mensagem, é exactamente relativo aos tipos de tramas apresentados (W, C, F, R, L, A ou M). Este parâmetro permite aos dispositivos receptores, perceber como deverão tratar o conteúdo seguinte, da mensagem recebida.

Depois do tipo de trama aparecem os dados, respectivo a cada tipo de trama, que serão apresentados posteriormente.

Quando uma trama está completa, antes de ser enviada, é sujeita a um algoritmo de cálculo de um CRC de 8 bits. O valor obtido por esse cálculo será anexo à trama, separado desta pelo caracter '%'. Este CRC permitirá, do lado do receptor, verificar se a trama recebida corresponde à trama que foi enviada e se não houve qualquer tipo de erro na transmissão dessa trama.

O caracter '|' é indicador do fim de trama. Tanto este caracter como o caracter '~' permitirão aos dispositivos conectados à rede, eliminar ruído que possa surgir nas comunicações, recebendo apenas tramas iniciadas pelo caracter '~' e terminadas pelo caracter '|'.

Dados da Trama de Alarmes – Tipo W

Os dados existentes na trama tipo W estão, genericamente, representados na figura 3.4. O dado seguinte ao tipo de trama, numa trama W, pode ser o pedido para leitura de alarmes, se for enviado pelo Mestre Central ou pelos Mestres aos seus respectivos Escravos, ou então a resposta, a esse pedido por parte dos Escravos, resposta esta que pode indicar, no caso de ocorrência de algum alarme, o tipo de alarme ocorrido, ou no caso contrário a sua não existência.

---	#	Tipo_Alarme / Pedido	#	Descrição_Alarme	#	Valor_Alarme	%	---

Figura 3.4 - Dados da Trama de Alarmes - Tipo W

Assim, no parâmetro "tipo de alarme", poderá surgir:

- Um 'R' – indicativo do pedido de mensagem por parte dos Mestres, ou ainda da resposta a esse pedido, se não existirem alarmes;
- Um 'T' – que indicará a ocorrência de um alarme de *threshold*, que ultrapassou os limites estabelecidos para os sensores de temperatura;
- Um 'E' – anunciando a ocorrência de um alarme, também, de *threshold*, que ultrapassou os limites estabelecidos para os extensómetros;
- Um 'A' – denunciando a ocorrência de um alarme, também, de *threshold*, nos valores medidos pelos acelerómetros;
- Um 'S' – relativo ao alarme de sensores em mau estado, por apresentarem medições sem sentido;
- E finalmente um 'B' – que aparecerá quando uma das baterias dos sensores estiver, quase descarregada.

A descrição do alarme, é relativa a uma especificação mais pormenorizada, do alarme que possa ter ocorrido e aplica-se apenas, na resposta ao pedido de alarmes. Ao surgir, num alarme de *threshold*, o carácter 'U' neste parâmetro, indica que, o sensor especificado atrás ultrapassou o limite superior do *threshold* estabelecido. Se o caracter verificado for um 'L', o Mestre percebe que foi ultrapassado o limite inferior, este caracter aplica-se ainda para os alarmes de pouca carga nas baterias dos Escravos. Para alarmes tipo 'S', o valor que surgirá neste parâmetro, será relativo ao identificador do sensor que possa estar com algum problema. No caso de ser um pedido para leitura de alarmes, o caracter presente neste parâmetro é o mesmo que no parâmetro anterior, ou seja, um 'R'.

Por fim, o valor do alarme é relativo ao valor inteiro verificado numa medição efectuada, ou, no caso das baterias, o valor, em percentagem, da sua carga. Para o caso das tramas de pedido para leitura de alarmes, este valor não tem significado.

Assim, uma possível troca de mensagens deste género pode ser a seguinte: o Mestre Central envia a trama "~1#99#5#W#R#R#0%89|" que significa que quer perguntar ao Mestre 1 se ocorreu algum alarme nos seus Escravos, repare-se que no ID do Escravo aparece o valor 99, que não tem qualquer significado, visto a pergunta ser direccionada apenas ao Mestre 1.

Se o Mestre 1 pretende-se ler alarmes do Escravo 2, a trama poderia ser "~1#2#5#W#R#R#0%42|".

Uma resposta possível no primeiro caso, se o Mestre 1 não possui-se nenhum alarme, seria "~1#99#5#W#R#R#0%89|", que é exactamente igual à pergunta. Já uma trama de resposta do Escravo 2 ao seu Mestre se tivesse medido, por exemplo 31ºC, valor que ultrapassa um possível *threshold* de 30ºC, poderia ser "~1#2#5#W#T#U#31%220|".

Dados da Trama de Sincronização dos Relógios – Tipo C

A trama de sincronização dos relógios dos vários componentes é sempre enviada, pela Unidade de Comando do sistema, a todos os Mestres e estes, são responsáveis por envia-la a todos os seus Escravos. Quando envia a mensagem, o Mestre Central fica à espera de uma confirmação, de todos os Mestres, da sincronização dos seus relógios e dos relógios dos seus respectivos Escravos.

Para sincronizar os relógios é necessário, enviar uma data e uma hora actual, de referência, que será a do computador central. Assim, os dados presentes nestas tramas (figura 3.5) são ordenadamente: ano; mês; dia; hora; minutos e segundos do computador central.

---	#	Ano	#	Mês	#	Dia	#	Hora	#	Minutos	#	Segundos	%	---

Figura 3.5 - Dados da Trama de Sincronização dos Relógios - Tipo C

A sequência de troca de mensagens para sincronização dos relógios de todos os dispositivos do sistema será a seguinte: primeiramente o Mestre Central envia ao Mestre 1 uma trama deste género "~1#99#5#C#2011#3#22#18#19#14%74|" e para o Mestre 10 uma idêntica apenas com diferença no ID do Mestre "~10#99#5#C#2011#3#22#18#19#14|". O Mestre 1, ao receber esta trama sincroniza o seu relógio e envia uma trama de sincronização aos seus escravos 1 e 2, respectivamente: "~1#1#5#C#2011#3#22#18#19#14%201|" e "~1#2#5#C#2011#3#22#18#19#14%201" e o Mestre 10, envia para o seu Escravo uma trama idêntica ao seu escravo: "~10#10#5#C#2011#3#22#18#19#14%170|". Os Escravos, ao receberem esta mensagem sincronizam de imediato os seus relógios e se essa tarefa for realizada com sucesso, enviam uma trama de resposta ao Mestres, exactamente igual à trama da ordem enviada por estes Mestres. Os Mestres, por sua vez, se conseguirem sincronizar os seus relógios e se receberem uma confirmação da sincronização de todos os seus Escravos, respondem ao Mestre Central com uma mensagem igual à que este lhe tinha envia anteriormente.

Dados da Trama de Definição das Funções e Activação de Sensores de um Escravo – Tipo F

Esta é a trama que permitirá ao operador escolher, quais os sensores que pretende ler, bem como de que forma o pretende fazer. Permitir-lhe-á assim, colocar os Escravos, em modo de leitura contínuo, em modo de leitura durante um intervalo de tempo e ainda, colocar um Escravo em "*sleep mode*", ou seja, desligando-se por completo para que não consuma energia da sua bateria. Sendo então a trama composta pelos dados apresentados na figura 3.6.

Esta trama será enviada pelo Mestre Central, directamente a um dos Escravos do Sistema e é esperada pelo Mestre uma trama igual à enviada para confirmação da definição.

---	#	Função	#	Data_Ini	#	Data_Fim	#	Hora_Ini	#	Hota_Fim	#	Número_sensores	:	ID_Sensor1	:	ID_Sensor2	...	%	---

Figura 3.6 - Dados da Trama de Definição das Funções e Activação de Sensores nos Escravo – Tipo F

No parâmetro Função, pretende-se definir a função pretendida para o Escravo, assim, neste campo da trama pode surgir: continuousA; continuousS; intervalA; intervalS; ou sleepA. Que indicam que o Escravo deve funcionar, respectivamente: em modo de leitura contínuo para todos os seus sensores; em modo de leitura contínuo para um número de sensores específico, que será definido em seguida; em modo de leitura de todos os sensores durante um intervalo de tempo a definir em seguida; em modo de leitura durante um intervalo de tempo a definir e para um número de sensores também a definir em seguida; ou se deve entrar em "sleep mode".

Seguido da função, para o caso de se pretender ler sensores num intervalo de tempo são definidas todas as unidades de tempo necessárias: Data de Inicio e Data de Fim, no formato aaaammdd, por exemplo 20110505; e a Hora de Inicio e Hora de Fim, no formato hhmmss, por exemplo 123400.

Depois dos parâmetros temporais, seguem-se os parâmetros referentes aos sensores. Importa aqui relembrar que o número de sensores activos, não pode ser superior a 4, como já foi explicado anteriormente, portanto a trama "continuousA", não se aplica para o Escravo 1 que possui seis sensores, assim, este terá que ser sempre definido pela função "continuousS". Os parâmetros da trama F referentes aos sensores, corresponder ao número de sensores que se pretende activar, seguido dos identificadores desses sensores, parâmetros estes, que são separados pelo caracter ":".

Repare-se então, que quando é definida a função "continuousA" ou "sleep", não há necessidade de enviar mais nenhum parâmetro; quando se pretende a função "continuousS" não é necessário enviar os parâmetros temporais; e quando se pretende a função "intervalA", não se enviam os componentes respectivos aos sensores. Verificando-se assim, que uma mesma trama pode apresentar aspectos muito diferentes, o que implica um processamento um pouco mais elaborado dos dispositivos do sistemas.

Dois exemplos, deste tipo de tramas, podem assim ser: "~1#1#21#F#sleepA%241|" e "~1#1#21#F#intervalS#20110313#20110315#123040#204050#4:1:2:3:4%216|".

Dados da Trama de Leitura dos Valores dos Sensores de um Escravo - Tipo R

Esta trama é também, enviada directamente pelo Mestre Central, a um dos Escravos do sistema. Pretende-se com esta trama ler os valores dos sensores activos pela trama tipo F. Pois, como o sistema funciona num modelo Mestre – Escravo, o Escravo só enviará ao Mestre Central os valores dos sensores que estiverem activos mediante um pedido. Então, trama de tipo R é a trama responsável por esse pedido.

A trama de pedido enviada pelo Mestre Central, não possui quaisquer dados adicionais ao cabeçalho e ao final das tramas padrão. Uma possível trama de pedido para leitura de dados pode então ser: "~1#1#101#R%18|".

Por sua vez, a resposta ao pedido (figura 3.7) é um pouco mais elaborada.

---	Num_Sensores	#	ID_Sensor1	:	Result_Sensor1	#	ID_Sensor2	:	Result_Sensor2	...	%	---

Figura 3.7 - Dados da Trama de Resposta Leitura dos Valores dos Sensores – Tipo R

Nesta resposta aparece um primeiro valor, referente ao número de sensores que foram lidos, seguido dos ID's dos sensores e respectivos resultados, separados por ';' ou por '#', como representado na figura.

Assim uma possível resposta ao pedido do Mestre Central referido em cima pode ser: "~1#1#101#R#3#1:11#2:22#3:33%165|".

Dados da Trama de Leitura de Vários Valores consecutivos dos Sensores de um Escravo – Tipo L

Esta trama foi desenvolvida, essencialmente para leitura de valores de acelerómetros, pois, estes necessitam de ser adquiridos a uma frequência de 100 Hz, o que pode não ser possível de realizar ao utilizar a trama tipo R.

---	ID_Sensor	#	Índice_Mensagem	#	Num_Tramas	#	Num_Dados	%	---

Figura 3.8 - Trama de Leitura de Vários Valores consecutivos de um Sensor

Na figura 3.8, está apresentada a fracção de dados príncipal desta trama. Ela possui: um identificador do sensor que se pretende ler nesse escravo; um índice relativo à sequência de trocas de mensagens necessárias; o número de tramas; e o número de dados que serão enviados com os valores das medições, como ficará evidente mais à frente.

Então, a sequência de mensagens trocadas pelos dispositivos para realizar esta tarefa é a seguinte:

1. O Mestre Central envia uma primeira mensagem, a indicar a sua intensão de ler vários valores de um sensor, com o índice 1, por exemplo: "~1#1#4#L#6#1%120|";
2. O Escravo, responder-lhe-á com uma mensagem de índice 2, e com o número de dados que serão lidos, bem como, o número de tramas necessárias para enviar esses dados, por exemplo "~1#1#4#L#6#2#5#95%27|".Definiu-se que por trama só se poderiam enviar 20 dados, para não se obter pacotes de mensagens demasiado grandes e definiu-se ainda, um limite de dados de 200, para não ocupar demasiado a rede. Então, na trama de exemplo, é indicado ao Mestre que serão enviadas cinco tramas, contendo as primeiras quatro 20 valores e a última 15;
3. Quando o Escravo envia a mensagem de índice 2 passa a comportar-se como Mestre, nesse instante fica à espera de uma resposta igual à que enviou e só quando a receber é que executa as suas medições;
4. Recebida a resposta do Mestre Central pelo Escravo, este começa a medir os dados a uma frequência definida e no final envia-os, de forma consecutiva, da seguinte modo: "~dado1#...#dado20%CRC|", até à última mensagem: "~dado81#...#dado95%CRC|;
5. O Mestre Central quando recebe os dados todos, envia uma mesagem de confirmação com índece 3 no final, como por exemplo "~1#1#4#L#6#3%196|".

Dados das Tramas de Leitura e Mudança de Estado dos Actuadores de um Escravo – Tipo A e Tipo M

As tramas de leitura e mudança de estado dos actuadores (tipo A e tipo M) são muito idênticas, à excepção de que na primeira para realizar o pedido não é enviado o parâmetro Estado.

Estas tramas estão representadas na figura 3.9.

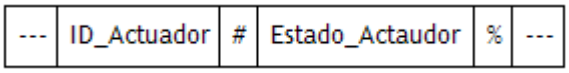


Figura 3.9 – Trama de Leitura e Mudança de Estado dos Actuadores de um Escravo – Tipo A e Tipo M

Um pedido de leitura, por parte do Mestre Central, pode então ser deste género: "~1#1#7#A#2%180|" e a resposta do Escravo: "~1#1#7#A#2#2%175|".

Para mudar o estado do actuador o Mestre pode enviar: "~1#1#8#M#2#2%204|" e fica a aguardar uma resposta igual da parte do Escravo.

Capítulo 4

Unidade de Interface, Comando e Controlo

Como já foi realçado no capítulo 3 (Arquitectura do Sistema), a Unidade de Interface, Comando e Controlo consistirá num computador central, onde será implementada grande parte do comando e controlo do sistema, assim como, a interface homem – máquina que permitirá ao operador aceder a todas as variáveis do mesmo.

Importa ainda relembrar a arquitectura de um sistema SCADA, presente na secção 2.2.2, onde se explica, que um sistema SCADA pode ser dividido em três componentes: a interface HMI, que permite ao operador o acesso às variáveis do sistema; o MTU, referente à unidade central do sistema, responsável pela recolha, processamento e controlo dos dados obtidos pelos sensores de campo, e ainda, encarregue de encaminhar as ordens, originárias de pedidos realizados pelo operador na interface, ao dispositivo correcto para a executar; e finalmente os RTUs, que são responsáveis por recolher os valores dos sensores e envia-los para as unidades superiores, bem como, executar, a partir dos seus actuadores, as ordens recebidas dos elementos de comando.

Conjugando estes dois últimos parágrafos pode-se, mas não de forma totalmente correcta, depreender que as funcionalidades da interface HMI e do MTU serão executadas nesta Unidade de Interface, Comando e Controlo. De facto, a interface HMI, será de total responsabilidade desta unidade, todavia, as tarefas do MTU serão repartidas entre esta e a Unidade de Comunicação e Controlo Inteligente, a qual será apresentada no capítulo posterior. Então, mais concretamente nesta Unidade, serão realizadas as tarefas relativas à recolha, processamento e armazenamento dos dados, mostrando esses dados de forma correcta ao operador e avisando por correio electrónico os responsáveis pela estrutura. Desta forma, estes dados podem ser considerados actuadores de informação, ou seja, tarefas de Controlo. Por último, será também desta Unidade que se realizará o Comando de todas as operações (leitura sensores, leitura de alarmes, sincronização, etc.) possíveis neste sistema. Por realizar todas estas tarefas aqui referidas, esta Unidade é assim designada de Unidade de Interface, Comando e Controlo.

As outras funções relativas aos MTUs da arquitectura SCADA, como a função de comunicação entre a Interface (Unidade de Interface, Comando e Controlo) e os RTUs (Unidades de Processo) e ainda, recolha e armazenamento provisório de dados relativos a

alarmes ocorridos, serão realizadas na Unidade de Comunicação e Controlo, como ficará evidente no capítulo 5.

4.1 - Implementação da Unidade de Interface, Comando e Controlo

Esta unidade foi desenvolvida utilizando linguagens de programação da família da .NET, uma *framework* desenvolvida pela Microsoft, que consiste, numa plataforma única de desenvolvimento e execução de sistemas de aplicações. E tal como já foi referido, uma das grandes vantagens desta plataforma é, permitir que todo o código nela gerado, possa ser executado em qualquer computador que a possua.

A Microsoft possui acompanhada à *framework* .NET, um elevado número de ferramentas de software que permitem a analise, design, programação, teste e desenvolvimento de aplicações nesta própria plataforma. Sendo que, a linha de produtos Microsoft Visual Studio .Net integra muitas dessas ferramentas para desenvolver aplicações [20]. Esta é portanto, um software que permite o desenvolvimento de aplicações, em todas as linguagens de programação da *framework* .NET, incluindo as pretendidas C# e ASP.NET. Por tudo isto, será este o software a utilizar, na versão Microsoft Visual Studio 2010.

Por conseguinte, a Unidade de Interface, Comando e Controlo pode ser divida em três módulos: a Interface Homem - Máquina (HMI), desenvolvida em C#; o Web Site, implementada em ASP.NET e uma Base de Dados remota (MySQL), que será manipulada directamente pela interface.

4.1.1 - Implementação da Interface Homem – Máquina

Comunicações

A base de funcionamento da interface HMI implementada é a comunicação, entre esta e as outras Unidades do sistema. Existe no entanto, no Visual Studio, uma biblioteca que permite a utilização directa de portas série (RS-232) do computador. Incluiu-se então, uma porta série à interface que permitiu a comunicação da mesma com o exterior. Mas a utilização de uma porta série implica um conjunto de configurações necessárias na mesma, para garantir o seu bom funcionamento. Os principais parâmetros a configurar são: o número da porta série do computador pretendida; a largura de banda das transmissões dos dados; o número de bits por dado; o número de bits de fim "*stop bits*", elemento indicativo de fim de uma trama e a paridade dos dados. Todas estas configurações são importantes, principalmente porque a forma como forem aqui definidas, deve que ser igual à configuração dos dispositivos que lhes serão conectados, que no caso são as *gateways* RS-232 – CAN, para que a comunicação se realize correctamente.

Definiu-se assim, de forma definitiva, o número de bits por dados de 8 bits, o número de "*stop bits*" de 1 e a inexistência de controlo de paridade nos dados.

Relativamente ao número da porta RS-232 utilizada e à largura de banda, estas serão definidas dinamicamente pelo operador, através de uma pequena janela da interface, representada nas figuras 4.1a e 4.1b. Aqui o número da porta série deverá ser introduzido pelo operador e quanto à velocidade de transmissão, o operador poderá escolher um valor,

em bits por segundo, dentro das seguintes opções: 4800, 9600, 19200, 38400, 57600 e 115200, como está representado na figura 4.1b.

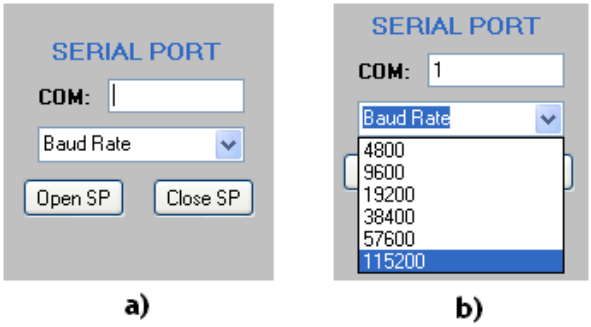


Figura 4.1 - Interface para Configuração da Porta Série

Configurada a porta série do computador que permitirá a comunicação à interface, podem agora realizar-se as comunicações entre esta e os seus Escravos. O envio de mensagens pela porta série na interface é muito fácil, é necessário apenas utilizar um comando para essa tarefa. No entanto, a recepção não é tão linear assim, seria também muito fácil se o programa estivesse em ciclo, à espera que uma mensagem fosse recebida nessa porta, mas, esta solução não é de todo a indicada, pois o programa ficaria bloqueado à espera dessa mensagem e não seria possível fazer mais nada na interface. Deste modo, a linguagem C# permite duas soluções para este problema: utilizando o conceito de *threads*, ou utilizando o conceito de interrupção por eventos.

O conceito de *threads* é relativo a um processo que é executado em paralelo com o processo principal, mas que podem estar incluídos num mesmo programa, ou seja, um mesmo programa pode conter o seu processo principal que corre normalmente e ainda, um ou mais *threads*, que são outros processos a correr nesse mesmo programa mas, de forma paralela ao processo principal, não influenciando, a não ser que sejam definidas para isso, o seu fluxo normal. Por sua vez, o conceito de interrupção por eventos é baseado na geração de eventos pela ocorrência de algum acontecimento externo, o que obriga o computador a deixar de realizar as tarefas de processamento que estava a realizar, para tratar esse evento. Por exemplo, clicar num botão de uma interface C#, é para o programa, um evento externo e pode esse acontecimento, desencadear um conjunto de tarefas ilimitadas. Por conseguinte, para a porta série, a biblioteca do Visual Studio, permite que seja considerado um evento a recepção de algum caracter na porta série, o que não acontece por exemplo com os *Pipes*, que serão utilizados para a comunicação com o Web Site que será (explicada mais à frente), onde é só possível, verificar a recepção de uma mensagem recorrendo a *threads*. Então, por ser relativamente mais simples de implementar e por implicar menos processamento ao computador, a solução utilizada para a porta série, será a de interrupção por eventos.

Desenvolveu-se também um procedimento para o tratamento genérico, do envio e recepção de mensagem por RS-232 na interface (figuras 4.2). Assim, todas as mensagens são enviadas com um identificador de mensagem diferenciável, o qual pode ser qualquer número inteiro do intervalo de 1 a 200. Para isso, existe uma variável global no programa que representa o ID das mensagens, que no inicio da execução do programa é inicializada a 0, antes do envio de alguma mensagem é incrementada e é, posteriormente, inserida na mensagem, no local definido pelo protocolo apresentado em 3.2.3. Previamente ao processo

de incrementação, é verificado se o seu valor é inferior a 200 e em caso negativo, em vez da variável ser incrementada é devolvida a 1. O valor máximo de 200 foi definido, para que não se ultrapasse o valor relativo máximo possível para um byte (255) e desta forma não sobrecarregar as comunicações.

Existirá ainda, uma lista (*array*) de mensagens, de tamanho igual ao número de mensagens "em comunicação" no sistema, ou seja, de tamanho igual ao número de mensagens enviadas, subtraído pelo número de mensagens recebidas e pelo número de mensagens que foram assumidas como não recebidas por *timeout*. As mensagens serão acedidas nesta lista, a partir do seu identificador.

Na figura 4.2, está representado o processo que se desenvolveu para o tratamento genérico das mensagens enviadas e recebidas pela porta série. Neste processo, a interface ao verificar que ocorreu um evento para o envio de uma mensagem *M*, executa de forma sequencial, o seguinte conjunto de procedimentos: é inicializada a 0 uma variável, relativa ao número de *timeouts* ocorridos desde que a mensagem foi enviada primeira vez; a mensagem é enviada; é guardada pelo seu ID na lista de mensagens "em comunicação"; e ainda antes de se regressar ao procedimento geral do sistema, é dada ordem a um temporizador para que comece a contar o tempo.

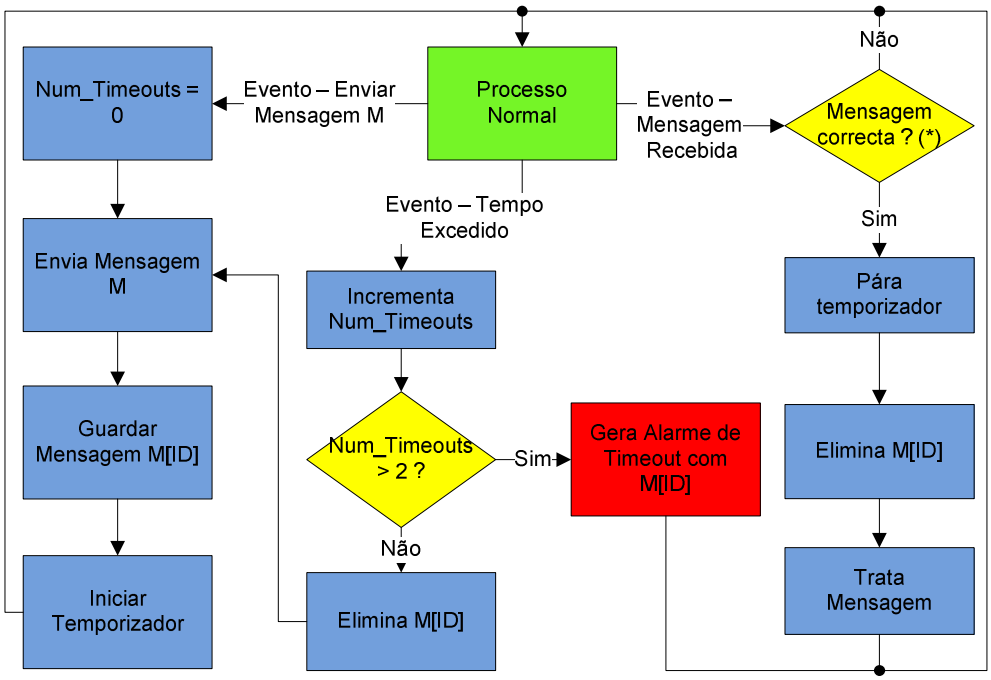


Figura 4.2 - Processo de Envio e Recepção de Mensagens na Interface pela Porta Série

Depois de enviada a mensagem, pode suceder uma de duas situações possíveis: ou é recebida a resposta a essa mensagem, assuma-se que apenas uma mensagem recebida apenas é assumida como resposta se passar a condição de "mensagem correcta"; ou o temporizador chegou ao fim da contagem, normalmente de 5 segundos, à excepção de tramas longas em que têm um tempo de *timeout* de 30 segundos. Na primeira situação, a sequência de tarefas a executar é, respectivamente: parar a contagem do temporizador; eliminar a mensagem enviada com o ID respectivo da lista e tratar essa mensagem, tratamento este, que será detalhado mais à frente. Na segunda situação, são executadas as seguintes rotinas:

incremento do número de *timeouts* que ocorreram desde o primeiro envio da mensagem; e posteriormente, verifica-se se esse número é superior a 2, ou seja, se é o terceiro *timeout*. Se a avaliação for verdadeira, é activado um alarme de *timeout* e é apresentado ao operador em que mensagem se deu o alarme, recorrendo ao ID dessa mensagem. Se a avaliação for falsa, é apagada a mensagem da lista e é enviada uma nova mensagem, idêntica à anterior, mas com um novo identificador.

Como já foi dito anteriormente, uma mensagem só é considerada uma mensagem recebida, se for aprovada por um conjunto de processos de verificação da sua credibilidade para o sistema. O primeiro desses processos é a verificação do primeiro caracter da mensagem recebida, se não for o caracter '~', definido no protocolo como caracter de inicio de trama, a mensagem é automaticamente ignorada. Em seguida, é calculado o CRC da mensagem recebida e é comparado com o CRC em anexo da mesma mensagem, se os valores forem diferentes, assume-se que ocorreu um erro de transmissão e a mensagem é ignorada. Finalmente, depois de aprovada a mensagem nos dois testes anteriores, é verificado se existe alguma mensagem com o identificador de mensagem que esta apresenta e, apenas mediante esta aprovação, é que a mensagem é tratada pelo sistema. Deve aqui ser realçado, que tanto neste como para todos os elementos que utilizem este protocolo, é possível receber p+elo menos duas mensagens numa mesma trama, ou seja, receber uma trama iniciada por '~' e seguida da mensagem pretendida, mas em vez de fechar a mensagem com um '|' é recebido um outro '~' na mesma trama, indicando o inicio de uma nova mensagem e apenas no final desta nova mensagem (se forem 2) é que aparece o caracter '|'. No entanto esta característica não foi muito utilizada neste sistema.

No intervalo de tempo entre o envio de uma mensagem e a recepção da sua resposta, ou a verificação, por timeout, dessa não recepção, podem ser enviadas outras mensagens sem que o processo descrito seja afectado. Isto porque, a cada mensagem é associado um só temporizador e também uma só variável de contagem de *timeouts* ocorridos.

Comunicações com Web Site

Para a interface comunicar com o Web site, ou mais correctamente, para o servidor do Web Site comunicar com a interface, que estão alocados no mesmo computador, são utilizados *Pipes*. Estes *pipes* são secções do Windows de memória partilhada, que permitem a diferentes processos implementados num mesmo computador comunicarem entre si, acedendo para tal, a estas secções de memória. Um processo criador de um *pipe* é um servidor *pipe*, e um processo que se conecta a um *pipe* é um cliente *pipe*. Quando é criado por um processo do sistema, um servidor um *pipe* e é conectado a esse *pipe* um processo cliente, estes podem comunicar entre si. Esta comunicação é baseada no conceito de memória partilhada, isto é, um dos processos associados a um *pipe* escreve informação nessa memória e o outro processo lê essa informação na memória.

Então, a comunicação entre o Web site e a interface será realizada segundo o modelo de cooperação Cliente – Servidor, comportando-se o Web site, mais especificamente o servidor do Web site, como Cliente e a interface como Servidor do modelo. Assim, a iniciativa de iniciar uma comunicação é sempre do Web site (cliente) que faz pedidos ao servidor, o qual lhe responde apenas a esses pedidos.

No entanto, a interface não pode estar bloqueada à espera destes pedidos e como já foi referido, esta comunicação, do ponto de vista da interface, ao contrário da comunicação pela

porta série, não pode ser iniciada por interrupções, sendo então, necessária a utilização de uma *thread* para a sua implementação. Esta *thread* está ciclicamente, a verificar se foi criado, pelo site, um servido *pipe*. E só muda de estado, ou seja, só avança o seu processamento quando este *pipe* é criado. Na figura 4.3, está representado, sob a forma de fluxograma, o processo que se desenvolveu para esta thread.

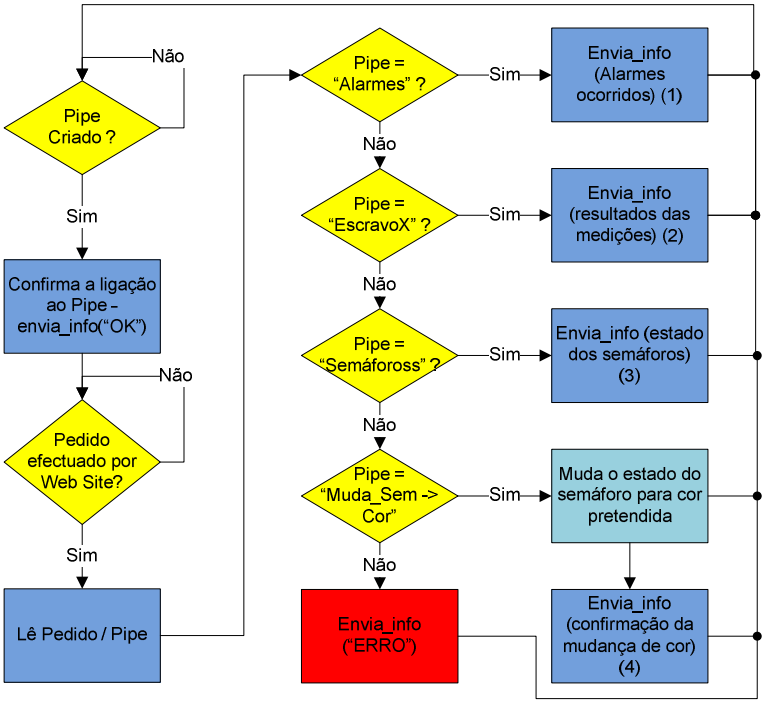


Figura 4.3 – Processo de Comunicaçã entre Interface e Web Site

Neste fluxograma, observar-se que a *thread* está inicialmente bloqueada até que um *pipe* seja criado, sendo que, esta criação ocorrerá por iniciativa do Web Site. Por conseguinte, quando é verificada a existência de um *pipe* específico, a interface escreve nessa memória partilhada a mensagem "OK", que indica ao Web Site a possibilidade deste, a partir deste instante, comunicar com a interface. A interface (servidor) fica neste momento, à espera que lhe seja efectuado algum pedido lhe pelo Web site. Quando isso acontece, o pedido é processado e é enviada a resposta adequada a esse pedido. Sendo que podem ocorrer um dos seguintes pedidos:

- Para leitura dos alarmes, neste caso a interface (servidor) envia os alarmes que possam ter ocorrido, seguindo o protocolo seguinte: "alarm1#alarm2#...#alarmN";
- Para leitura dos sensores dos Escravos, aqui a interface envia os valores da última medição que efectuou aos sensores, se a efectuou, seguindo o protocolo: "resultado_sensor1#resultado_sensor2#...#resultado_sensorN";
- Para leitura dos estados dos semáforos, onde é enviada a mensagem: "estado_semáforo1#estado_semáforo2";
- Para mudar o estado do semáforo, sendo este é o pedido mais complexo, pois implica o envio do estado do semáforo pretendido para o sistema, por parte da interface e só quando recebe uma confirmação dessa execução é que responde ao Web site, com uma mensagem de confirmação, ou de falha na operação desejada;

- Por fim, se não for recebido nenhum dos pedidos anteriores, o servidor envia uma mensagem de erro (“ERROR”).

Estado Global do Sistema

Um dos grandes requisitos para esta interface é de, a partir dela, ser possível rapidamente verificar o estado global da ponte, bem como, na eventualidade de ocorrência de uma anomalia verificar rapidamente o local onde esta ocorreu.

Na figura 4.4, está demonstrada a forma como se solucionou na interface este requisito. Então, de forma bem visível é possível observar logo em cima o estado global da ponte, neste trabalho, considera-se que a ponte está instável sempre que ocorre um alarme, seja ele qual for. Para além disso, como neste sistema existem apenas dois Mestres, pode considerar-se que só existem dois pilares na ponte, como tal, o sinóptico escolhido não representa mais do que esse valor de suportes. Neste sinóptico, será assinalado o local de ocorrência de alarmes, que pode ser no Mestre 1 ou no Mestre 2, com uma bola vermelha e com uma bola verde assinalando-se os locais da ponte considerados estáveis. Assim, na figura 4.4a pode ser observada um exemplo em que a ponte se encontra estável nos dois Mestres, já figura na figura 4.4b está representada uma situação em que alguma anomalia ocorreu no Mestre 2. Uma outra situação possível a ocorrência de um alarme nos dois Mestres, nesse caso seria ilustrada uma bola vermelha, no local de existência de cada um deles.

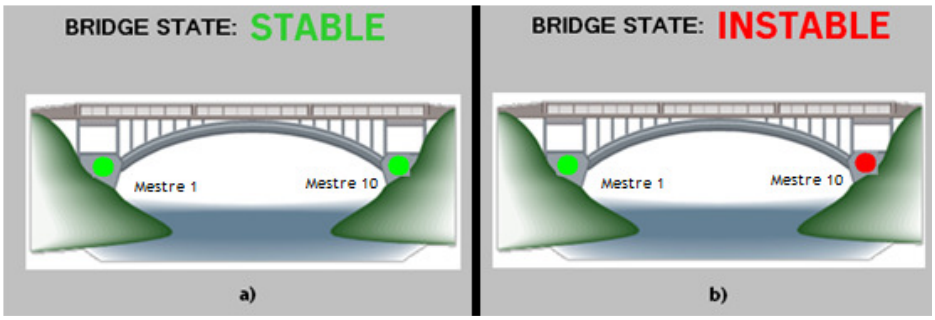


Figura 4.4 - Ilustração do Estado do Sistema - a) Estável e b) Instável

Lista de Alarmes

Um outro requisito, fundamental a esta interface é a apresentação de alarmes ao operador. Portanto para detectar a possível ocorrência de alarmes no sistema, a interface deve enviar uma mensagem de pergunta de um possível aparecimento de alarmes, e a uma frequência alta, relativamente ao tempo real deste sistema, por exemplo, em cerca de 1 em 1 segundo. Essa mensagem, é enviada conforme o protocolo apresentado na secção 3.2.3 e seguindo o procedimento citado no inicio desta secção, na alínea das comunicações da interface.

Com o envio deste pedido é esperado uma resposta, que siga as normas do protocolo geral. Essa resposta, é depois de aprovada pelos métodos de verificação de erros já apresentados, sujeita a um conjunto de procedimentos tais como: verificação da sua origem, Mestre e Escravo respectivos; verificação da ocorrência ou não de um alarme e em caso afirmativo, qual o tipo de alarme ocorrido; ler a descrição do alarme e do seu valor de anexo associado. Com este processo é possível à interface, apresentar o alarme, de forma mais clarificada, do que a mensagem obtida no protocolo. Isto é, em vez de apresentar ao

operador, como por exemplo, a mensagem: "~10#10#63#W#A#U#65|" é lhe oferecida uma mensagem com a "tradução" da anterior como: "*Passed Upper Limit of Acceleration in Master 10 - Slave 10 => measured value: 65g*". Este é um exemplo de uma mensagem de alarme por *threshold*, onde o limite superior estabelecido para o acelerómetro foi ultrapassado, se em vez de ultrapassado o limite superior fosse o inferior, surgiria na mensagem anterior no lugar de *Upper* o termo *Lower*. Se fosse outro sensor, em vez de Acceleration, apareceria *Extension* ou *Temperature*, para extensómetros e sensores de temperatura, respectivamente.

No caso de o alarme ser devido a sensores avariados, a frase a surgir seria, por exemplo: "*Malfunction of Sensor 2 - in Master 1 - Slave 2 => measured value: 100*", onde o ID do sensor é ajustado ao ID recebido na mensagem, bem como, a origem do alarme e o valor medido.

Já para a trama de aviso de fraca bateria de algum dispositivo, Escravo, pois os Mestres podem ser alimentados pela mesma alimentação que as *gateways*, a mensagem a apresentar será do tipo "*Battery in Master 1 - Slave 1 is Low => value*: 10%", onde os parâmetros ajustáveis são, como é previsível, a origem do alarme e a percentagem de bateria carregada.

Por fim, os alarmes de *timeout*, que são verificados localmente, aparecem com frases relativas à mensagem de falhas de envio, por exemplo, se a mensagem de alarmes não for correctamente enviada a descrição mostrada ao operador é "*Timeout ERROR!! - Reading Alarms form Masters*", existe, para além desta, uma mensagem diferente para cada situação de *timeout* possível como: numa falha da sincronização dos relógios; na definição do modo de leitura dos Escravos; para leitura de medições de sensores; na leitura de estados dos actuadores e ainda para mudança de estado dos actuadores. Nesta mensagem de alarme é ainda, nos casos aplicáveis, indicado o Mestre e o Escravo que não conseguiu responder à mensagem.

Na figura 4.5, pode-se observar a forma como os alarmes são apresentados ao operador na interface. Estes são mostrados numa tabela, na qual se vão inserindo, na forma ilustrada, os alarmes que vão ocorrendo. Repare-se que, quando a interface é arrancada a primeira vez, é indicado dado conta ao operador que tudo se encontra estável, até que os alarmes vão surgindo e completando a tabela há medida que vão sendo verificados.

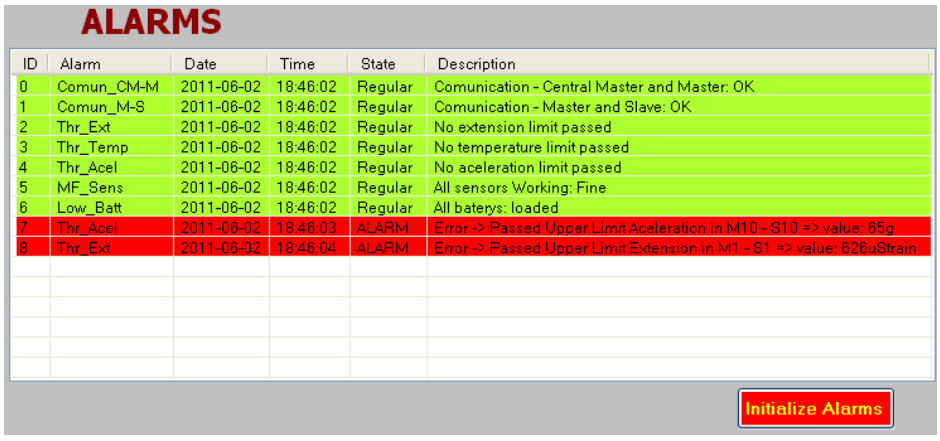


Figura 4.5 - Lista de Alarmes na Interface

Como já dito neste documento, não se espera que exista um operador 24h por dia a supervisionar a interface, portanto, se a ocorrência de um alarme ficar apenas registada na

interface, corre o risco de não ser observado a curto prazo. Para dar solução a esta situação, elaborou-se um método para envio de um email por cada alarme ocorrido, aos responsáveis pela obra, utilizando para tal uma biblioteca especifica do software Visual Studio. Neste caso prático utilizou-se um endereço de email de teste. O email enviado terá como Assunto (*Subject*): "*ALARM on the bridge!*" e no texto desse email, aparecerá uma descrição idêntica à utilizada na tabela da interface, bem como, a data e a hora do alarme sucedido.

Sincronização dos RTCs

A interface é o elemento do sistema responsável por sincronizar os relógios (RTCs) de todos os dispositivos do sistema. Para isso, foi desenvolvida, na interface, uma pequena aplicação (figura 4.6).

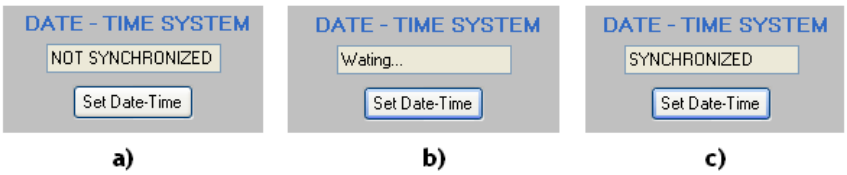


Figura 4.6 - Sincronização dos RTCs na Interface. a) Não Sincronizados, b) Apenas Mestre 1 Sincronizado e c) Todos Sincronizados

O algoritmo relativo ao processo de sincronização dos RTCs que se implementou está ilustrado na figura 4.7.

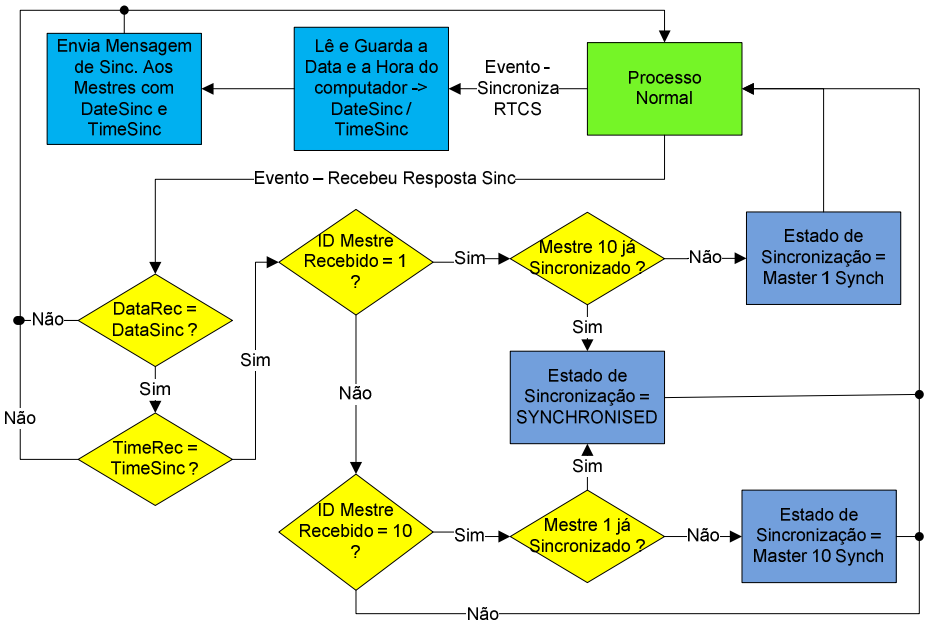


Figura 4.7 - Processo de Sincronização dos RTCs

A iniciativa de sincronizar os relógios é do utilizador da interface, ao premir o botão "*Set Date-Time*". Esta acção gera um evento no programa que o leva a fazer, ordenadamente as seguintes tarefas: leitura da Data e da Hora do computador onde está a ser corrida a interface e memorização desses valores; envio da mensagem de sincronização a todos os

Mestres do sistema, que têm a responsabilidade de sincronizar os seus Escravos associados, com a mesma hora e data. Quando estas operações terminam, aparece na caixa de texto do estado da sincronização, a informação "*Wating…*" (figura 4.6b). Na recepção de uma resposta ao pedido de sincronização, a interface verifica numa primeira análise se a data e a hora recebidas são a data e a hora que enviaram na sua mensagem de sincronização. Caso este teste seja bem sucedido, o programa verifica, em seguida, se o Mestre obteve a resposta e seguidamente verifica se o outro Mestre já confirmou a Unidade de Comando a sua sincronização. Se sim, a mensagem escrita na caixa de texto é "*SYNCHRONISED*" (figura 4.4c), se não, é registado na caixa de texto, que o Mestre respectivo está sincronizado.

Esta última avaliação, neste sistema com apenas dois Mestres é muito pequena, basta verificar se o outro Mestre já se encontra ou não sincronizado, mas, num sistema com mais Mestres teria que ser realizada para todos eles, isto é, num sistema com *n* Mestres esta avaliação teria que ser realizada *n*-1 vezes para os diferentes Mestres. Note-se ainda que, só é recebida uma resposta de sincronização de cada vez, isto por imposição da rede CAN, então, mesmo no caso que as gateways recebam uma confirmação dos Mestres ao mesmo tempo, é realizada uma arbitragem a essas mensagens e só é enviada uma de cada vez, este processo será explicado no capítulo seguinte.

No tempo de espera de recepção da confirmação, por parte dos Mestres, da sincronização dos seus relógios internos, o programa geral corre normalmente, podendo ser neste instante realizados outros pedidos/ordens ao sistema.

No caso de a operação não ser bem sucedida, ao fim das três tentativas, realizadas pelo algoritmo apresentado no inicio desta secção, aparecerá a mensagem "*SYNC_FAIL*".

Leitura dos Sensores

A interface HMI do sistema deve, como foi referido no sistema, permitir ao operador escolher quais os sensores que pretende monitorizar e de que forma o pretende fazer, em modo contínuo ou durante um intervalo de tempo, e ainda, facultar ao operador a possibilidade de adormecer os dispositivos de processo (Escravos), para poupar energia das suas baterias. Para isso, desenvolveu-se as aplicações gráficas representadas nas figuras 4.8a e 4.8b.

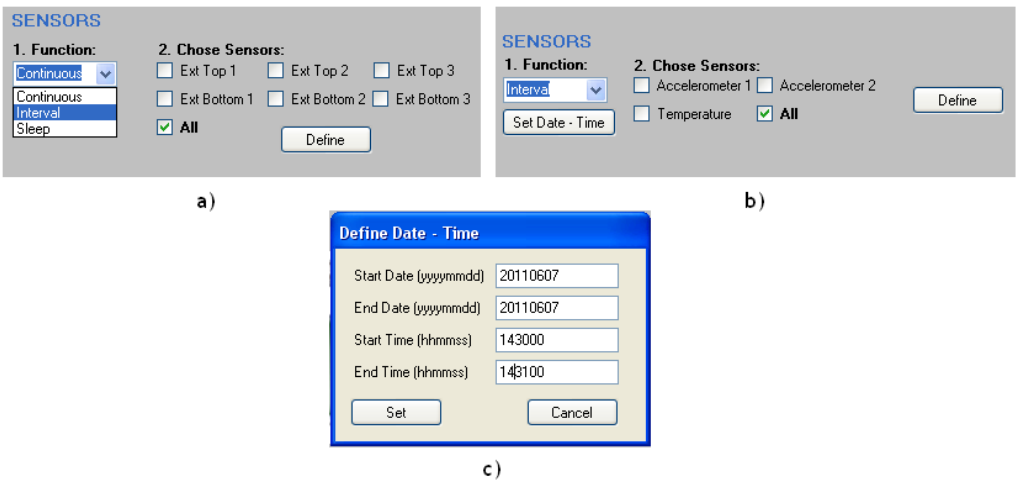


Figura 4.8 - Definição nos Sensores para Leitura na Interface: a) Escravo 1, b) Escravos 2 e 10 e c) Janela de definição do Intervalo de Tempo de Leitura

Na figura 4.8a, está representada a aplicação relativa ao Escravo 1 e em 4.8b a aplicação relativa aos Escravos 2 e 10, que são iguais uma vez que, apresentam o mesmo número da sensores para cada tipo. Quando é seleccionada na caixa de múltipla escolha, a função de ler sensores num intervalo de tempo, surge na interface um botão com a inscrição: "*Set Date - Time*", como está representado em 4.8b. Se a função seleccionada for outra, este botão ficará oculto. Quando este botão é premido, aparece a janela representada em 4.8c, que permite ao operador definir o intervalo de tempo pretendido para efectuar a medição, no exemplo ilustrado na figura, o operador pretende ler os sensores no dia 07-06-2011, durante um minuto e com inicio às 14 horas e 30 minutos.

Definidos, pelo operador, todos os sensores que se pretendem ler, não esquecendo que no máximo podem ser quatro, bem como, a função com que se pretende efectuar essa leitura, o mesmo pode clicar no botão "*Define*" para que a ordem seja enviada ao escravo respectivo, que lhe confirmará a aceitação da ordem caso consiga executa-la.

Depois de definidos todos os parâmetros mencionados em cima e recebida a confirmação do Escravo dessa definição, a leitura dos valores dos sensores pode ser iniciada. Nas figuras 4.9a e 4.9b, pode ser visualizada a forma como são representados os resultados das medições dos sensores na interface. Esta leitura é realizada pela interface, utilizando a trama de leitura (tipo R), apresentada no protocolo de comunicação geral do sistema.

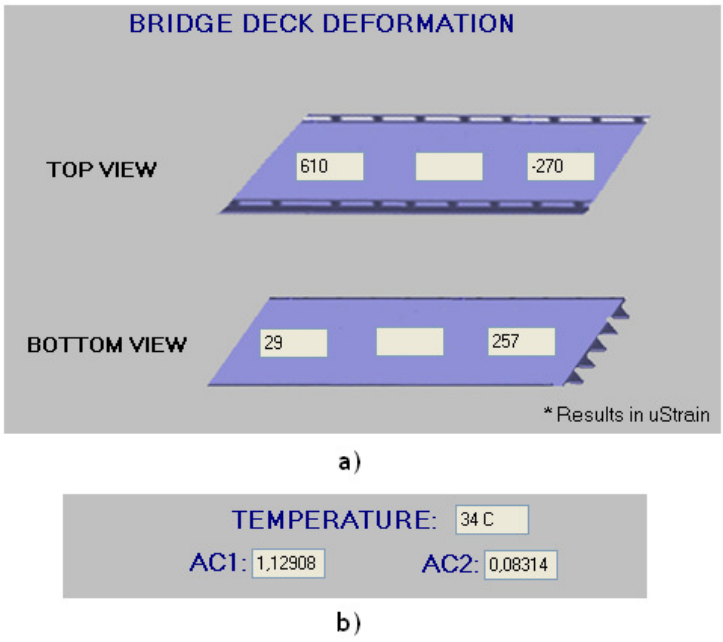


Figura 4.9 - Ilustração dos valores lidos nos dispositivos de processo na Interface: a) Escravo 1 e b) Escravos 2 e 10

Leitura de Vários Valores Consecutivos de um Sensor

O algoritmo implementado para receber vários valores consecutivos, utiliza as tramas tipo L do protocolo de comunicação definido, e é relativamente mais complexo que o necessário para a situação anterior, aliás como se deve facilmente compreender. Esse algoritmo está representado na figura 4.10.

Verifica-se, pela figura anterior que o processo normal, quando é interrompido por um evento de leitura com vários valores consecutivos de um sensor, envia de imediato uma

mensagem com pedido desses valores (trama L com o índice 1), especificada no protocolo. Finalizado esse pedido, o processo normal contínua a decorrer com normalidade, não ficando assim bloqueado à espera da resposta dos Escravos, que neste caso em particular pode demorar alguns segundos. Quando o processo normal é depois interrompido pela resposta ao seu pedido (trama L com índice 2), ocorre o processamento da trama recebida para que se guarde o valor relativo ao número de tramas, com o máximo de 20 valores, que serão recebidas, bem como o número total de medições/dados.

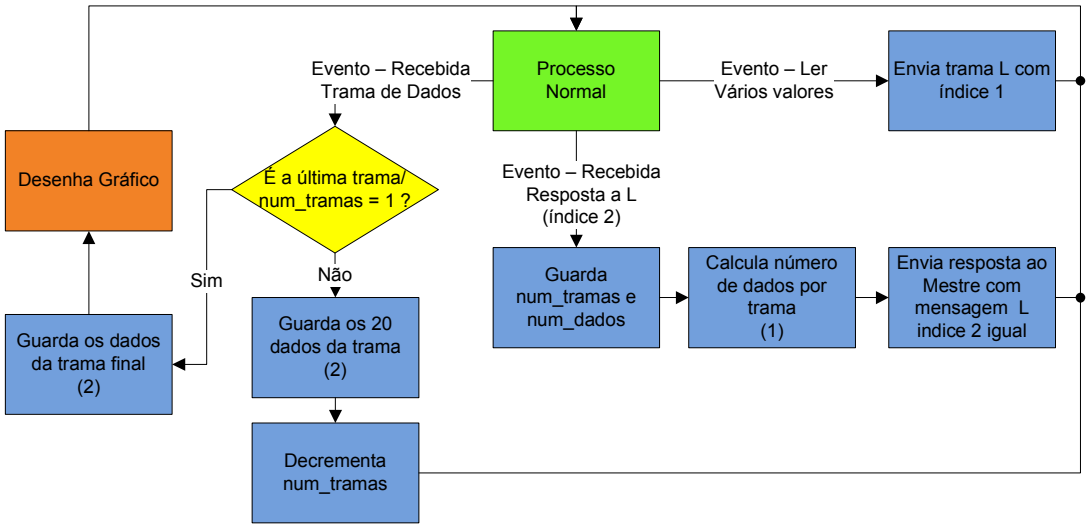


Figura 4.10 - Processo para leitura de vários valores consecutivos das medições dos sensores

A acção seguinte, diz respeito ao cálculo do número de dados que serão recebidos em cada trama, mais especificamente, o número de tramas completas (tramas com 20 valores) e o número de dados da última trama (menor ou igual a 20). Este processo de cálculo está representado na figura 4.11a. Aqui, o primeiro cálculo realizado é relativo ao número de tramas completas que se esperam receber, dividindo o número de dados total pelo número de dados máximo numa trama (20 dados). O valor inteiro do resultado desta divisão, indica o número de tramas completas. Obtido esse valor, é comparado o número de tramas que se esperam receber, com o número de tramas completas, e se esses valores forem iguais, percebe-se que a última trama é completa, ou seja, tem 20 dados de medições. No caso do número de tramas completas ser menor que o número de tramas esperadas, obtém-se o número de dados da última trama, pelo resto da divisão entre o número de dados esperados e número de dados máximo por trama. Realizados estes cálculos, o processo contínua normalmente, passando em seguida para a tarefa de envio da resposta ao Escravo, confirmando a recepção dos dados da trama L de índice 2, regressando novamente ao processo normal. Quando recebe uma trama de dados, o algoritmo desenvolvido avalia se essa é a última trama, se não for, guarda os 20 dados da trama, se for a última trama, guarda o respectivo número de dados já calculado.

A forma para obtenção dos valores da trama de dados está representada no fluxograma da figura 4.11b. Neste, percebe-se que a primeira tarefa a realizar é a obtenção do primeiro valor da trama de dados, em seguida é decrementado o número de dados da trama que está a ser lida, para o caso das tramas completas é de 20 valor, e para a última trama é um valor

menor ou igual a 20. Verifica-se depois de decrementado, se o número de dados ainda presente na trama é nulo, ou seja, se já foram ou não lidos todos os dados da trama em causa. Se não, é lido o seguinte valor presente na trama e o processo repete-se, se sim assumem-se como obtidos todos os valores da trama e o processo continua normalmente.

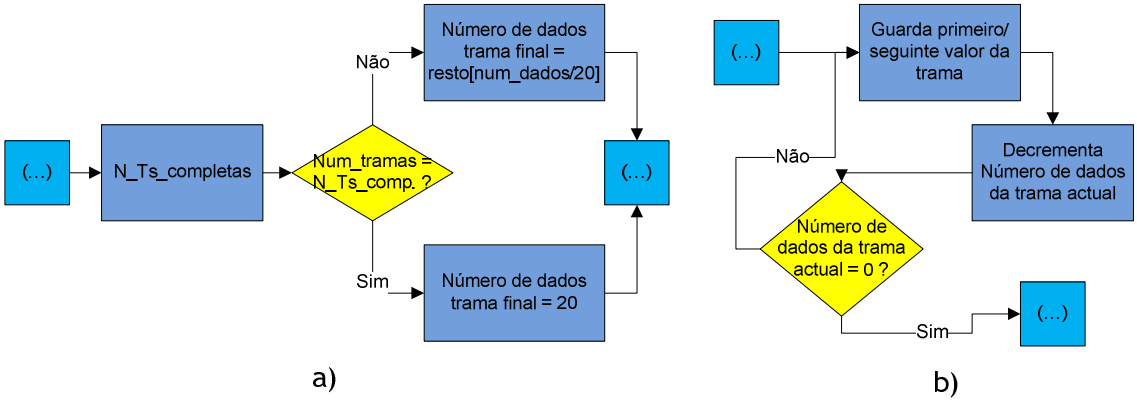


Figura 4.11 - Sub-processos de leitura de vários valores consecutivos das medições dos sensores

Quando são adquiridos todos os dados, a interface desenha um gráfico ilustrativo de todas essas medições (figura 4.12).

Esta forma de leitura, foi desenvolvida para obter os resultados das medições de acelerómetros, que devem adquirir os seus dados a 100 Hz, portanto, a escala de tempo apresentada no gráfico exemplificado (figura 4.12), que é definida pela interface, mostra os dados de 10 em 10 milisegundos, no eixo das abcissas. Estando no eixo das ordenadas o valor obtido das medições em g. Esta frequência não foi, no entanto, a frequência utilizada nos ensaios que se apresentarão no capítulo 7, onde se conseguiu, apenas, uma frequência de cerca de 10Hz.

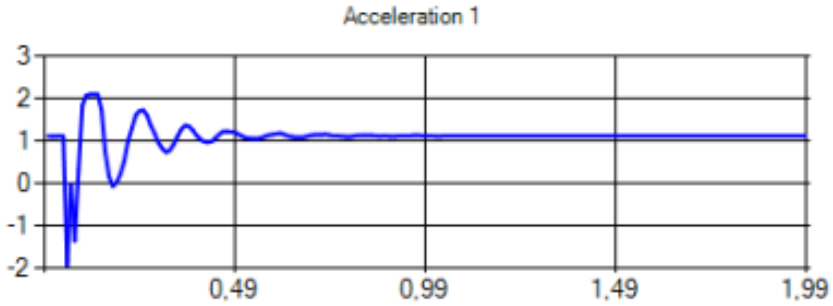


Figura 4.12 - Exemplo de gráfico demonstrativo dos resultados obtido na leitura de vários valores consecutivos de um acelerómetro

Leitura e Mudança de Estado dos Actuadores

Como foi referido nos requisitos, esta interface deve permitir ao operador ler e mudar o estado dos seus actuadores, que neste caso particular, são dois semáforos.

Deste modo, na interface o utilizador pode ler o estado de cada um destes semáforos, clicando para isso no botão "*Read State*" do semáforo que pretende, e enquanto estiver a ser realizada a leitura desse respectivo semáforo, aparecerá a mensagem "*Reading…*", como se

pode verificar na figura 4.13a. No final de uma leitura bem sucedida desse estado, o semáforo representativo ficará com a cor referente ao estado em que se encontra.

Para alterar o estado do semáforo, o operador, pode numa caixa de escolhas múltiplas (figura 4.13b), definir a cor que pretende e seguidamente premir o botão "*Set*". Enquanto a troca das mensagens para mudança de estado do semáforo respectivo é realizada, aparecerá a mensagem "*Setting…*" e quando a tarefa for concluída, se não ocorreram erros, o semáforo representativo ficará com a cor visível pretendida.

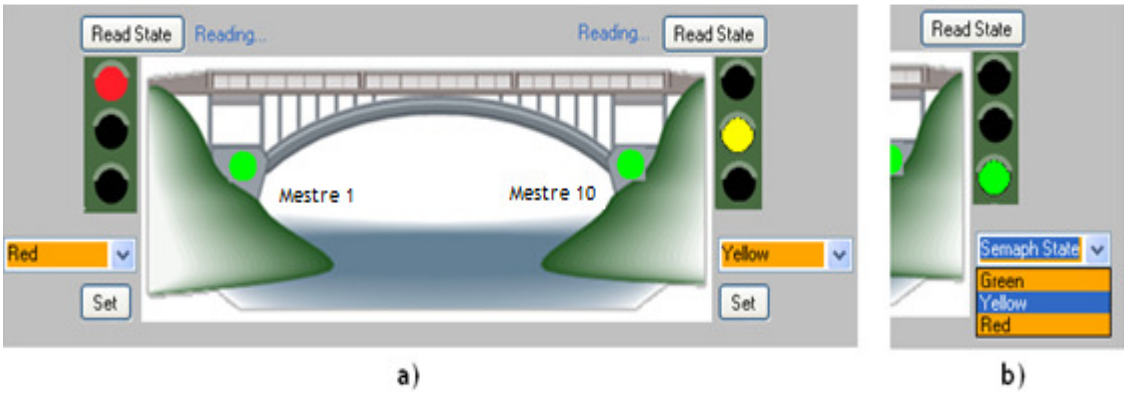


Figura 4.13 - Leitura e Mudança de Estado dos Semáforos na Interface

Relatórios

Foi adicionada, ainda à interface, a capacidade para esta produzir relatórios diários, às 23 horas e 59 minutos de cada dia, com uma síntese global dos alarmes, medições e mudanças de estado dos actuadores do sistema. Relatório este, que será depois de produzido, enviado por email, como anexo, aos responsáveis pela obra, da mesma forma que são enviados os alarmes.

Esta funcionalidade foi desenvolvida, recorrendo à ferramenta gratuita *iTextSharp*, que pode ser facilmente integrada no Visual Studio. Com esta ferramenta é possível criar documentos em PDF, que será o formato utilizado para estes relatórios. Um exemplo com um pequeno excerto de um relatório está ilustrado na figura 4.14.

Neste relatório exemplo é possível visualizar, o aspecto dos relatórios diários produzidos pela interface, bem como, a representação dos dados nos mesmos.

Estes relatórios apresentam, seguido de um pequeno título e subtítulo, a lista de alarmes ocorridos no sistema no dia de produção do relatório e posteriormente são apresentadas umas tabelas relativas à média, máximo e mínimo dos valores medidos nos sensores de cada Escravo, bem como, o número de vezes que o semáforo esteve, nesse dia, em cada um dos estados. Neste excerto, apenas estão representados as grandezas relativas ao Escravo 1, mas no seu original estão de seguida representados os valores dos restantes Escravos. E note-se ainda que, os dados apresentados neste relatório são apenas ilustrativos, nada tem a ver com medições realmente efectuadas.

Em suma, os dados utilizados para produzir este relatório são retirados da base de dados do sistema.

REPORT 2011-05-10

Structure Report of 2011-05-10

ALARMS

ID	TIME	TYPE	DESCRIPTION
12	01:33:18	Thr_Temp	Error -> Passed Upper Limit Temperature in M10 - S10 => value: 110K
13	01:34:06	Thr_Ext	Error -> Passed Lower Limit Extension in M1 - S1 => value: 20units

SLAVE 1

SENSOR	AVG	MAX	MIN
Extensometer1_1	1,0000 uStrain	1 uStrain	1 uStrain
Extensometer1_2	2,0000 uStrain	2 uStrain	2 uStrain
Extensometer1_3	3,0000 uStrain	3 uStrain	3 uStrain
Extensometer1_4	4,0000 uStrain	4 uStrain	4 uStrain
Extensometer1_5	5,0000 uStrain	5 uStrain	5 uStrain
Extensometer1_6	6,0000 uStrain	6 uStrain	6 uStrain

ACTUATOR	STATE	NUMBER OF TIMES
Semaphore1_1	Green	0
Semaphore1_1	Yellow	2
Semaphore1_1	Red	2

Figura 4.14 - Exemplo de um Excerto de um Relatório Diário em PDF

Aspecto Final

Com todos as aplicações, descritas anteriormente, juntas, obteve-se o aspecto final ilustrado na figura 4.15.

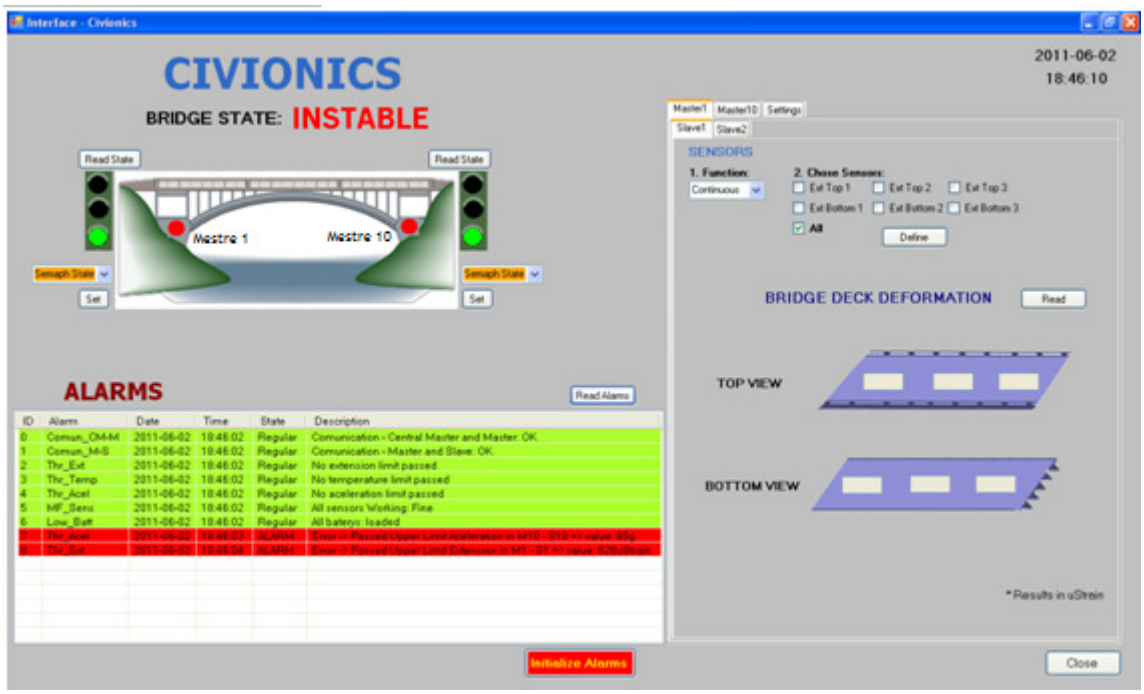


Figura 4.15 - Aspecto final da Interface

Como se pode verificar, no Estado global da ponte, o sinóptico e a lista de alarmes estão sempre visíveis ao operador. Já os Escravos e as configurações básicas (porta série e sincronização dos relógios) são acedidos por separadores, do lado direito da interface.

Na figura 4.16, está representado o separador relativo ao Escravo 1. O separador do Escravo 2 (igual ao Escravo 10) e o separador das definições básicas estão representados nas figuras 4.16a e 4.16b, respectivamente.

Figura 4.16 - Restantes Separadores da Interface

Uma funcionalidade que se projectou, mas não foi aplicada ao sistema, foi a capacidade de este autonomamente enviar aos seus Mestres e Escravos, pedidos para leitura de alarmes, bem como, realizar pedidos de leitura das grandezas medidas pelos Escravos quando o operador definir a função de ler continuamente, ou ler num intervalo de tempo, esses dados. O algoritmo desenvolvido para esta função é relativamente simples e está representado pelo fluxograma da figura 4.17.

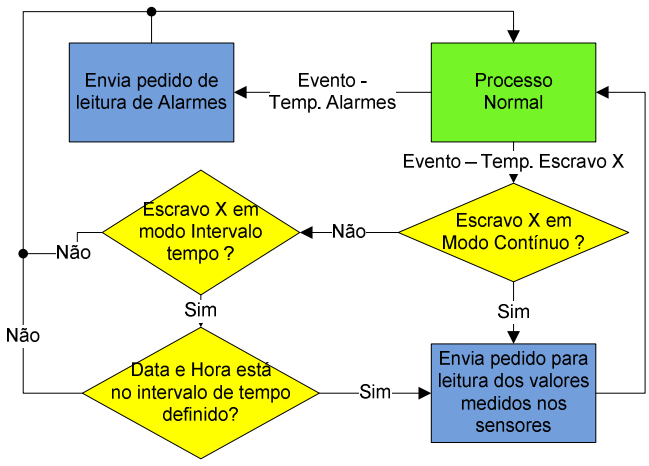


Figura 4.17 - Processo de funcionamento autónomo da interface

O funcionamento deste processo passaria por num intervalo de tempo definido, por exemplo de 1 em 1 segundo, enviar uma trama para pedido de leitura dos alarmes, bem como num outro intervalo de tempo, que pode ser um pouco superior ao anterior, enviar um pedido de leitura dos valores medidos nos sensores a cada um dos Escravos. Pedido esse que, só deveria ser efectuado mediante algumas restrições, tais como, o Escravo dever-se encontrar-se em modo de leitura contínua, ou então, em modo de leitura num intervalo de tempo, caso a data e a hora a que o evento de leitura desse Escravo ocorreu, estiver dentro do limite de tempo que o operador definiu previamente.

A justificação para este processo não ter sido implementado deve-se ao facto de este sistema ser utilizado apenas para alguns testes laboratoriais e não numa estrutura real única situação em que seria mesmo essencial. Para realizar os ensaios necessários, as mensagens são enviadas recorrendo a botões implementados na interface, os quais estão visíveis nas figuras 4.15 e 4.16.

4.1.2 - Implementação do Web Site – Interface Remota

O Web site, de monitorização do sistema, está dividido em 5 páginas distintas: a página de Autenticação; a página inicial; a página do Escravo 1; a página do Escravo 2; e a página do Escravo 10.

A primeira página apresentada no Web site é a página de autenticação (figura 4.18).

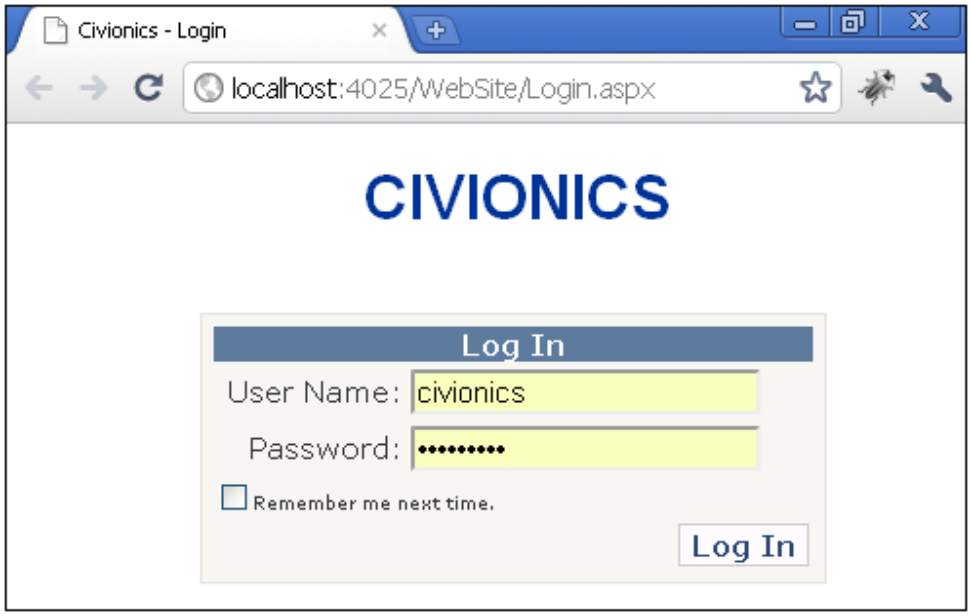


Figura 4.18 - Página de autenticação do Web site

Esta página de autenticação, permite ao utilizador autenticar-se para ter acesso à monitorização do sistema em si. O objectivo desta página, é o de impedir que intrusos que podem por em causa o bom funcionamento do sistema, tenham acesso a este. Assim o utilizador coloca o seu nome de utilizador e a sua palavra passe, os quais são convertidos em expressões MD5, recorrendo para isso a uma biblioteca do Visual Studio .Net. Estas expressões são seguidamente comparadas às expressões MD5 da palavra passe e do nome de utilizador, que foram previamente definidos, e que permitem o acesso ao Web site. Um exemplo deste processo é o seguinte: "*civionics*", que é o nome de utilizador que permite o acesso ao site,

tem como expressão MD5 “04-86-ED-19-D5-77-DE-93-8F-DB-18-F9-FD-DD-27-94”, então, se a palavra introduzida no nome de utilizador der origem a uma outra expressão MD5, que é como dizer que não é a palavra “*civionics*”, o utilizador não será autenticado. Deste modo, se algum intruso estiver à escuta dos dados trocados entre o cliente e o servidor do sistema, os dados que apanha são as expressões MD5, não conseguindo assim perceber os dados originais (nome de utilizador e a palavra passe), utilizados pelo utilizador para fazer a autenticação.

Quando o utilizador se consegue autenticar, passa directamente, para a página inicial do Web site (figura 4.19). Aqui, é lhe reportada de forma clara o estado global da estrutura, nesta figura exemplo está estável, portanto, o fundo da caixa de texto encontra-se verde, se estivesse instável a caixa de texto estaria vermelha. Ainda, à imagem da interface HMI do sistema, é apresentado um sinóptico da ponte, com apenas dois pilares, onde será possível visualizar o local de ocorrência de uma eventual anomalia no sistema, neste caso está tudo a funcionar correctamente, pois os dois pilares têm uma bola verde sobre eles. Em cada extremidade da imagem da ponte surge também, um semáforo representativo do estado desse actuador, que pode ser remotamente alterado utilizando para tal a caixa de múltipla escolha, situada por baixo de cada semáforo, onde se escolhe o estado pretendido e clica-se em seguida em botão “*Set*” para confirmar essa escolha.

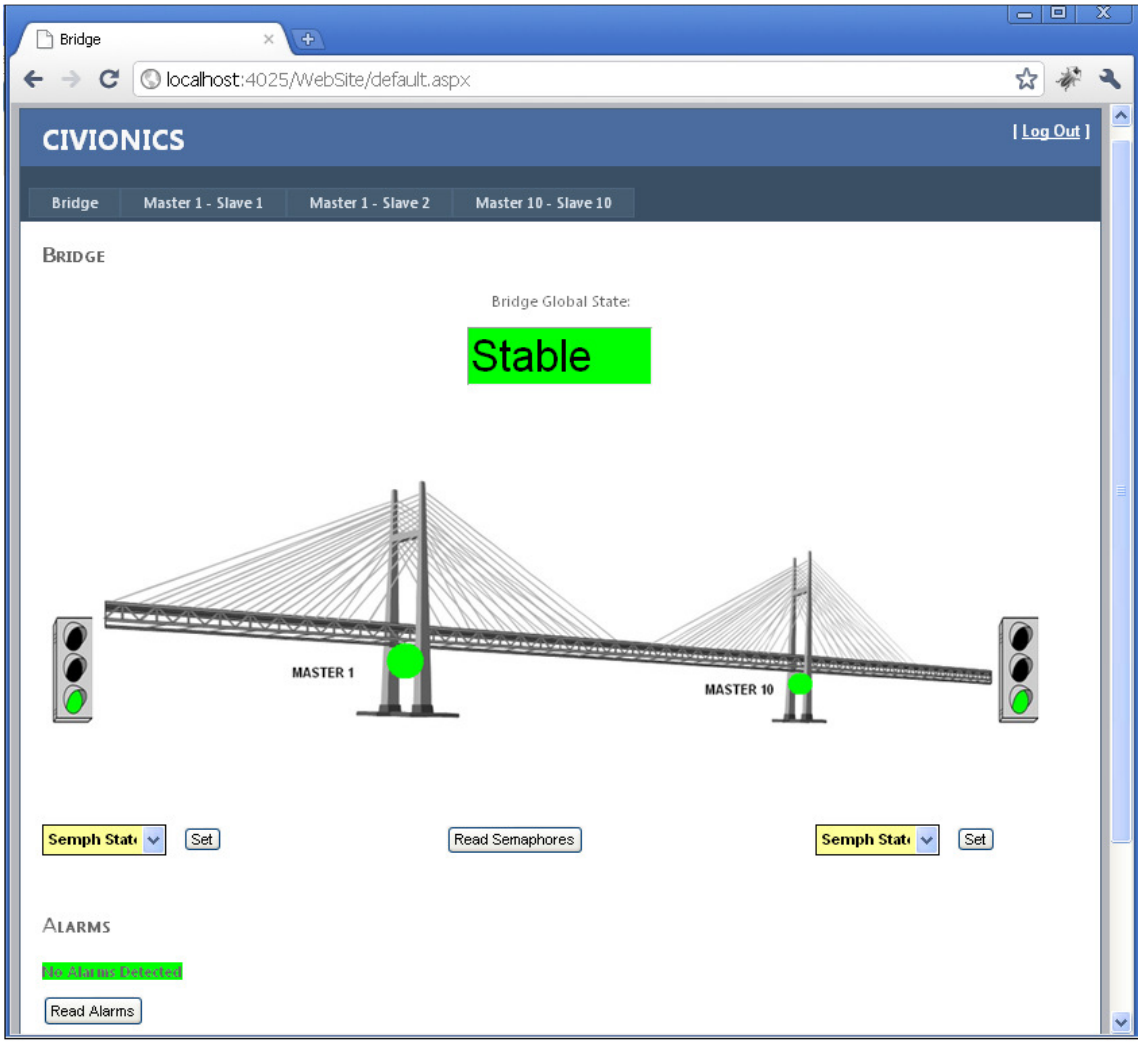


Figura 4.19 - Página Inicial do Web Site

Na parte de baixo desta página aparece ainda, a lista dos alarmes detectados na interface, neste exemplo, não existe nenhum, caso algum alarme fosse detectado, ele surgiria numa tabela de forma idêntica à da interface, como se pode verificar na figura 4.20. Estes alarmes podem ser lido, por intenção do utilizador ao clicar no botão "*Read Alarms*".

Adicionalmente, no cabeçalho do site é possível, alterar a página mostrada no Web site através dos Separadores: Brigde, que nos remete para a página inicial e os "*Master 1 – Slave 1*", "*Master 1 – Slave 2*" e "*Master 10 – Slave 10*" que nos mostram as páginas dos Escravo 1, Escravo 2 e Escravo 10, respectivamente.

Date:	Time:	Alarm:
2011_06_02	18:46:03	Error__>_Passed_Upper_Limit_Aceleration_in_M10__S10_=>_value:_65g
Date:	Time:	Alarm:
2011_06_02	18:46:04	Error__>_Passed_Upper_Limit_Extension_in_M1__S1_=>_value:_626uStrain

Figura 4.20 - Alarmes na página inicial do Web site

Nas páginas relativas aos escravos, está presente apenas uma imagem, na qual são ilustrados os locais da ponte onde os sensores "estão instalados", mas não se refere obviamente a uma ponte real, é apenas uma representação possível (figura 4.21 e 4.22).

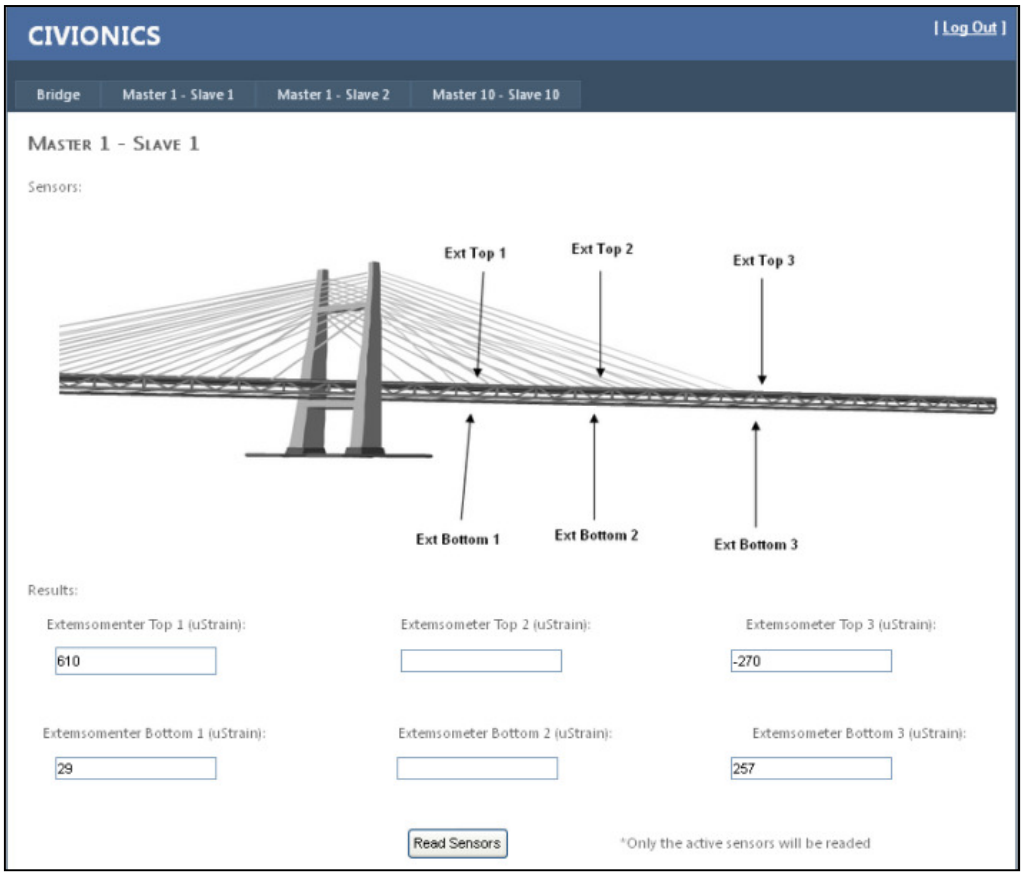


Figura 4.21 – Páginas Web - Escravo 1

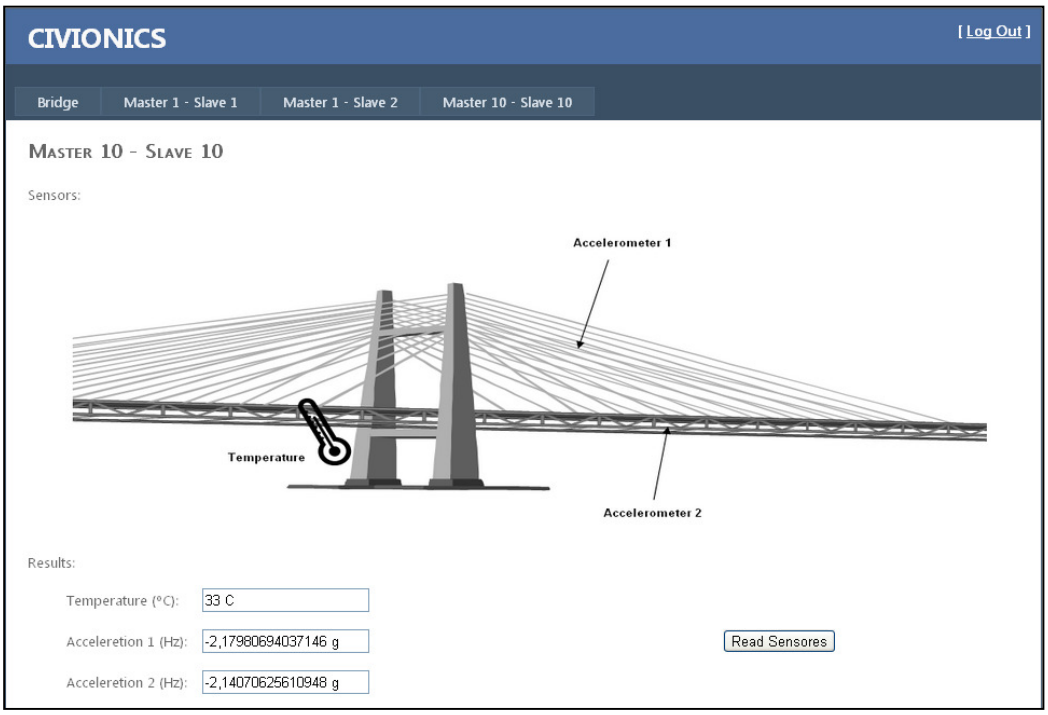


Figura 4.22 – Páginas Web - Escravo 2 e 10

Nestas páginas, estão ainda presentes umas caixas de texto referentes a cada um dos sensores dos Escravos, idêntico também, ao que acontece com a interface. Para além disso, existe um botão (*Read Sensors*), que permite fazer a leitura de todos esses sensores, quando o utilizador o desejar.

No entanto, este botão de leitura dos sensores, não realiza as mesmas funções que o botão da interface, ou seja, não envia uma mensagem ao Escravo respectivo a pedir que sejam realizadas medições pelos seus sensores. O que este botão faz, como já ficou mais ou menos evidente na secção anterior na parte de comunicação com o Web site, é pedir à interface que lhe envie o seu último valor lido. O mesmo verifica-se para qualquer outro dos botões existentes, à excepção do botão que permite alterar o Estado do semáforo, que pede à interface que comunique directamente com o processo para fazer essa alteração.

Relativamente ao carregamento de todas as páginas, excluindo a página de autenticação, é enviada uma mensagem à interface pedindo as respectivas informações para cada página; assim para o caso da página inicial é pedido o estado dos semáforos e os alarmes existentes e no caso das páginas dos Escravos, são pedidos os valores das medições respectivas. Portanto, sempre que a página é actualizada (*refresh* à pagina), são pedidas novas informações à interface, que poderão já ter sido alteradas desde a última vez, ou não.

Pegando nesta funcionalidade, facilmente se percebe que se o site for actualizado automaticamente, num intervalo de tempo definido, este tornar-se-ia praticamente um site dinâmico, de “tempo-real”. Então, foi exactamente isso que se implementou, para tal, utilizou-se um temporizador, que é uma das ferramentas AJAX disponibilizadas pelo Visual Studio, onde se lhe definiu o tempo e desenvolveu-se ainda, o programa para que sempre que o tempo do temporizador seja ultrapassado se realize o *refresh* da página em questão, actualizando assim os seus valores em “tempo-real” com uma frequência igual à do temporizador.

Definiu-se um minuto para este tempo de actualização, valor que é um pouco alto do ponto de vista da monitorização em tempo real, mas se fosse muito mais pequeno, poderia tornar-se incomodativo para os utilizadores a constante actualização da página, principalmente para ligações à internet mais lentas.

4.1.3 - Base de Dados

A base de dados é outra das partes integrantes do sistema de supervisão e controlo que se desenvolveu. Desta forma, utilizou-se para validação do sistema, uma base de dados do sistema MySQL, que permite a manipulação dos seus dados utilizando a linguagem SQL.

Todos os dados do sistema (alarmes; valores lidos dos sensores e ainda mudanças de estado dos actuadores) recebidos na interface, são ainda, por ela, colocados nessa base de dados, para que possam ser acedidos em qualquer altura, como por exemplo, na criação do relatório diário.

Para dar coerência aos dados introduzidos neste sistema de armazenamento, utilizou-se o diagrama de classes representado na figura 4.23.

Repare-se no diagrama, que a base de dados do sistema possui sete classes, que podem também ser designadas de tabelas.

A classe de alarmes é isolada das restantes, e refere-se aos alarmes ocorridos no sistema. Nesta existem cinco colunas: o identificador (id), que é único para cada dado inserido; a data, referente à data de ocorrência do alarme; a hora, que se refere à hora de ocorrência do alarme; o tipo, referente ao tipo de alarme ocorrido (*timeout, threshold* nos extensómetros, bateria fraca, etc.); e uma descrição, onde é inserido um pequeno texto descritivo do alarme ocorrido. A figura 4.24 exemplifica a forma como estes alarmes são visualizados na tabela da base de dados.

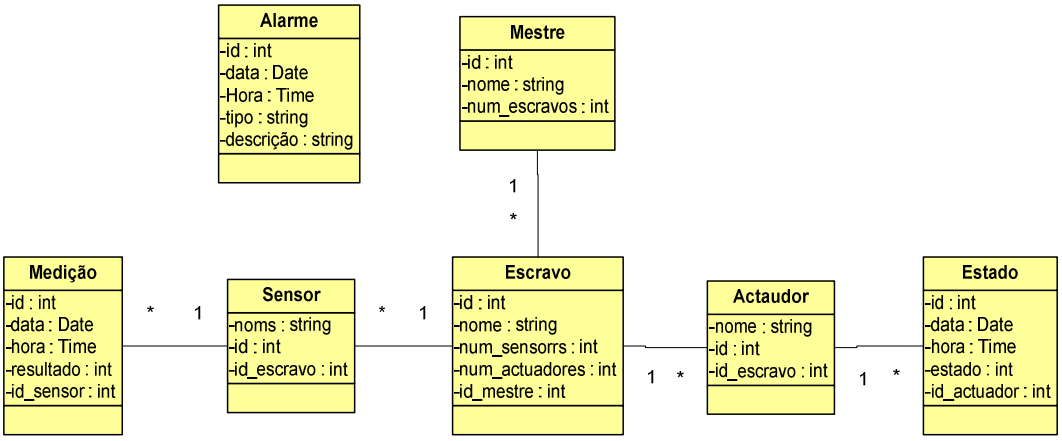


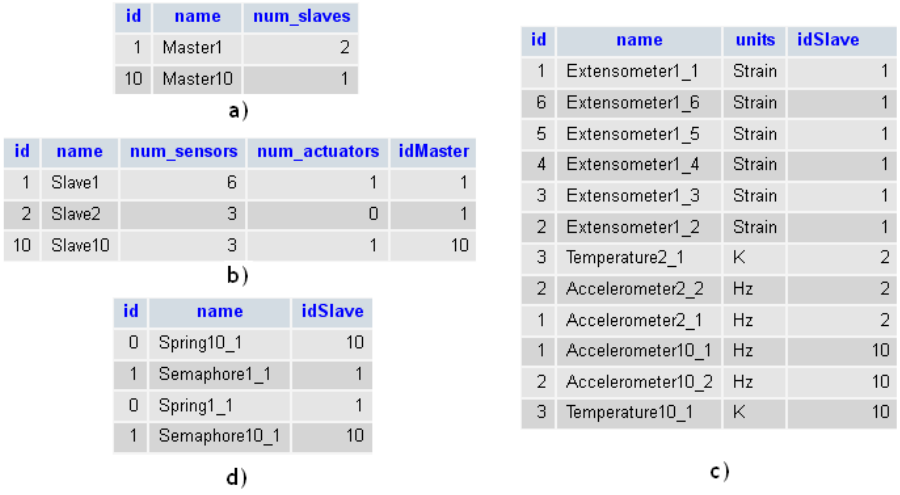
Figura 4.23 -Diagrama de Classes da Base de Dados

id	date	time	type	description
19	2011-05-14	17:14:03	Low_Batt	Error -> Battery in M10 - S10 is Low => value: 12%
36	2011-05-19	16:30:31	TimeOut	Reading Alarms form Masters
31	2011-05-19	16:10:38	TimeOut	Timeout ERROR!! - Synchronising Clocks

Figura 4.24 - Exemplos de dados na tabela de Alarmes da base de dados

As restantes seis classes do diagrama da figura 4.23 estão associadas entre si. Deste modo, percebe-se pelo diagrama, que a um Mestre, que representa na base de dados os dispositivos de comunicação e controlo inteligente do sistema, pertence um ou mais Escravos, que por sua vez possuem vários sensores e vários actuadores. Aos sensores serão associadas medições e aos actuadores serão associados estados.

Os dados das tabelas Mestre, Escravo, Sensores e Actuadores são fixos e estão representados nas figuras 4.25a, 4.25b, 4.25c e 4.25d, respectivamente.



id	name	num_slaves
1	Master1	2
10	Master10	1

a)

id	name	num_sensors	num_actuators	idMaster
1	Slave1	6	1	1
2	Slave2	3	0	1
10	Slave10	3	1	10

b)

id	name	idSlave
0	Spring10_1	10
1	Semaphore1_1	1
0	Spring1_1	1
1	Semaphore10_1	10

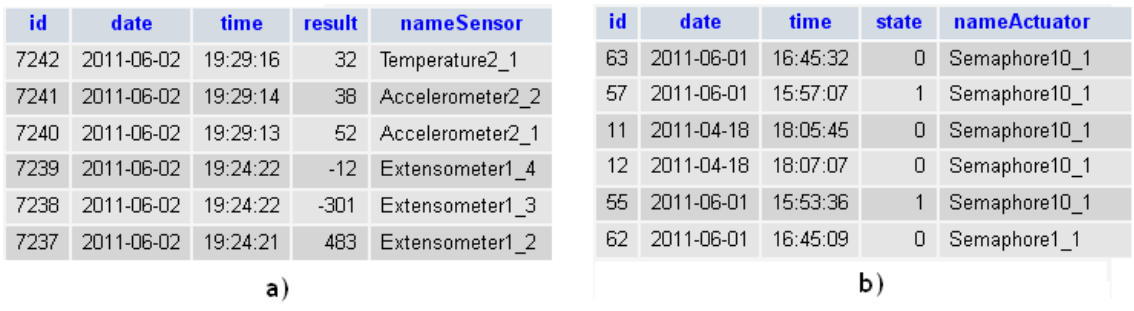
d)

id	name	units	idSlave
1	Extensometer1_1	Strain	1
6	Extensometer1_6	Strain	1
5	Extensometer1_5	Strain	1
4	Extensometer1_4	Strain	1
3	Extensometer1_3	Strain	1
2	Extensometer1_2	Strain	1
3	Temperature2_1	K	2
2	Accelerometer2_2	Hz	2
1	Accelerometer2_1	Hz	2
1	Accelerometer10_1	Hz	10
2	Accelerometer10_2	Hz	10
3	Temperature10_1	K	10

c)

Figura 4.25 - Tabela de dados fixos: a) Mestre, b) Escravo, c) Sensor e d) Actaudor

Por sua vez, a tabela de Medição e a tabela de Estados, à semelhança da tabela de alarmes, são dinâmicas. Sendo que na primeira, serão registadas todas as medições realizadas pela interface, bem como todos os estados alterados e lido por ela. Alguns exemplos, completamente ilustrativos, estão registados nas figuras 4.26a para as medições e 4.26b para os estados. No que toca aos estados dos semáforos, são nestas tabelas registados em números, entenda-se portanto, que 0 representa a cor verde, 1 representa a cor amarela e 2 representa a cor vermelha do semáforo.



id	date	time	result	nameSensor
7242	2011-06-02	19:29:16	32	Temperature2_1
7241	2011-06-02	19:29:14	38	Accelerometer2_2
7240	2011-06-02	19:29:13	52	Accelerometer2_1
7239	2011-06-02	19:24:22	-12	Extensometer1_4
7238	2011-06-02	19:24:22	-301	Extensometer1_3
7237	2011-06-02	19:24:21	483	Extensometer1_2

a)

id	date	time	state	nameActuator
63	2011-06-01	16:45:32	0	Semaphore10_1
57	2011-06-01	15:57:07	1	Semaphore10_1
11	2011-04-18	18:05:45	0	Semaphore10_1
12	2011-04-18	18:07:07	0	Semaphore10_1
55	2011-06-01	15:53:36	1	Semaphore10_1
62	2011-06-01	16:45:09	0	Semaphore1_1

b)

Figura 4.26 - Exemplos de dados nas tabelas: a) Medição e b) Estado da base de dados

Capítulo 5

Unidade de Comunicação e Controlo Inteligente

As Unidades de Comunicação e Controlo, auxiliadas pelas gateways CAN – RS-232, realizam conjuntamente com as Unidades de Interface, Comando e Controlo, algumas das tarefas características de uma unidade MTU da arquitectura do sistema SCADA, como já foi referido no capítulo anterior. Mais especificamente, esta unidade permitirá a interacção entre o computador central do sistema e as unidades de processo, que contém os sensores e actuadores. É por esta sua capacidade, que no seu nome se lê Unidade de Comunicação.

O conceito de Controlo é normalmente, associado a um controlador que possui uma determinada função de transferência e que produz umas determinadas saídas, em função das entradas que observa. E é o que de certa forma se verifica nestas Unidades, pois estes dispositivos enviam autonomamente, num período de tempo reduzido, pedidos aos dispositivos de campo (Unidades de Processo) ficando à espera de uma resposta dos mesmos (à espera das entradas). Processando essas respostas, os dispositivos percebem se ocorreu algum alarme ou não, nas Unidades de Processo e guardam-no provisoriamente, até que a Unidade Central lhe faça um pedido de leitura de alarmes, ao que estes deverão responder (produzindo saídas) com os alarmes que contém neles guardados. Para além disso, estas Unidades podem receber pedidos para sincronização do seu e dos RTCs dos dispositivos de processo a ele associados, aos quais deve enviar uma mensagem de sincronização, esperando receber respostas confirmativas dessa execução, que são por eles interpretadas e compreendidas. Podendo assim, depois de confirmada a sincronização de todos os Escravos a ele associados, responder à Unidade de Interface, Comando e Controlo com uma aprovação da tarefa de sincronização completa. Uma outra tarefa de processamento algo complexa destas unidades, tem a ver com as mensagens de leitura de vários valores consecutivos de um sensor. Estas são responsáveis por adquirir, com frequência elevada, dos dispositivos de campo, o número de medições consecutivas definidas e envia-las em tramas de 20 dados no máximo à unidade central, este processo este, que será explicado mais à frente. Portanto, por realizar estas funções, de forma completamente autónoma e bem definida, pode considerar-se este dispositivo um Controlador Inteligente.

Pelos parágrafos anteriores, repara-se também que este dispositivo, no que diz respeito ao modelo base de cooperação do sistema, o modelo Mestre – Escravo, desempenha em

situações distintas ambas as funcionalidades. Comportando-se como Mestre dos dispositivos de processo, quando autonomamente, envia pedidos para leitura de possibilidades de surgimento de alarmes.

Por outro lado, funciona como escravo do computador central, quando este lhe realiza pedidos para leitura de alarmes aos quais os Mestres respondem apenas com os dados que têm guardados. Nas restantes situações, este dispositivo funciona apenas como "ponte de comunicação" entre o Mestre Central e os Escravos (Unidades de Processo), tendo somente o papel de direccionar a mensagem recebida do Mestre Central ao dispositivo de campo indicado, ou a todos, como no caso das mensagens de sincronização dos relógios internos.

No sistema de supervisão e controlo protótipo desenvolvido, a comunicação das Unidades de Comunicação e Controlo Inteligente com a Unidade Central, será realizada recorrendo a uma rede CAN-Bus e a comunicação com as Unidades de Processo, efectua-se mediante a utilização do protocolo de redes sem fios ZigBee 802.15.4, aliás como já ficou evidente no capítulo 3 referente à arquitectura do sistema. Então, neste capítulo, antes de se apresentar a implementação das Unidades de Comunicação e Controlo Inteligente, será realizado um estudo teórico relativo a redes de comunicação industriais em geral, pormenorizando as redes utilizadas no sistema. E será também, apresentada implementação destas no sistema desenvolvido.

5.1 - Tecnologias de Comunicação

Desde a primeira máquina automatizada baseada em componentes electromecânicos até às grandes instalações de várias máquinas que trabalham coordenadamente, houve sempre um domínio comum: a ligação entre as máquinas e o seu meio ambiente. Uma máquina isolada necessita sempre de informações do seu meio ambiente para trabalhar correctamente, precisa por exemplo de sinais de fim de curso, sinais de presença de objecto, sinais de medida, etc. Como estamos imersos num mundo onde tudo, ou quase tudo, se baseia na electricidade, percebe-se facilmente que uma forma cómoda de transmitir um sinal de um sensor para uma máquina será a partir de um sinal eléctrico, transmitido por um cabo eléctrico que os une.

Este conceito de comunicação entre máquina e sensor aplica-se também entre máquina e sistema de controlo, coordenação e gestão das empresas. Dito isto, é evidente a necessidade de coordenação entre todos estes elementos. As redes industriais são elementos fundamentais numa empresa, sendo responsáveis pela interacção entre todas as suas áreas constituintes de forma simples atempada e eficaz, contribuindo assim muito activamente para uma melhoria da qualidade e eficácia de uma empresa, bem como para o aumento dos lucros, através da redução dos custos de produção, distribuição e venda dos produtos.

Os requisitos de comunicação utilizados nos diferentes níveis hierárquicos de uma empresa, são muito distintos. Podem ser o número de equipamentos que comunicam entre si; o tamanho das mensagens; o tipo de tráfego, periódico (processos contínuos) ou aperiódico (processos discretos); a velocidade de transmissão; entre outros. Assim, é necessária a selecção do tipo de rede de comunicação a utilizar em cada uma das áreas da empresa, que individualmente consigam cumprir com eficácia as suas funções, originando um sistema global, que funcione o mais eficazmente possível. Pode-se então considerar também uma

estrutura hierárquica para o sistema de comunicação de uma indústria, dividida em 3 tipos de redes: redes de fábrica, redes de célula e redes de campo.

As redes podem ainda ser classificadas de acordo com a sua distribuição geográfica, como LANs, WANs e DANs [21]:

- *Wire Area Network* (WAN) – rede de tecnologias de informação (IT), que permite comunicação com o exterior e do exterior ao sistema (internet);
- *Local Area Network* (LAN) – rede de IT, com o propósito de possibilitar a comunicação local entre sistemas de controlo;
- *Device Area Network* (DAN) – rede de comunicações utilizada pelos dispositivos de campo do sistema.

5.1.1 - Redes de Fábrica

As redes de fábrica são as redes utilizadas nos níveis mais superiores da arquitectura hierárquica de uma empresa. São responsáveis pela comunicação entre as áreas de gestão e engenharia de uma empresa, ou seja, pela comunicação entre elementos do mesmo nível hierárquico. Devem ainda, fornecer serviços de comunicação com os níveis inferiores e o exterior, como serviços de acesso à internet ou e-mail. A comunicação com níveis inferiores realizada de forma descendente, baseia-se em comandos de ordens de fabrico e do seu escalonamento, já a comunicação em sentido contrário, centra-se em dados relativos ao comportamento do processo de fabrico, que permitam análise do mesmo, no sentido de o procurar melhorar. Como nos níveis hierárquicos em que estas redes estão inseridas, são utilizados equipamentos bastante complexos. As redes devem ser escolhidas de forma a poderem ser facilmente utilizadas por esses equipamentos, sabendo à partida que o número de equipamentos utilizados é bastante reduzido. Outras características deste tipo de redes são: a sua capacidade de transmissão de um volume de dados bastante elevado, na ordem dos *G/Mbytes*; a sua frequência de transmissão pode ser baixa, na ordem dos segundos; os tempos de resposta também não têm necessariamente que ser muito elevados, podendo ser superiores a 1 segundo; a comunicação é normalmente aperiódica (por eventos); a sua cobertura geográfica deve ser muito abrangente; a largura de banda elevada (na ordem dos G/Mbits por segundo); e devem ainda, permitir a partilha de informação com outras aplicações (de voz, vídeo, etc.). Um tipo de comunicação utilizada nestes níveis que respeita todas estas restrições, é comunicação baseada em redes Ethernet com pacotes TCP/IP.

5.1.2 - Redes de Célula

As redes de célula situam-se no centro desta hierarquia, tendo como principal função a interligação dos dois níveis a si adjacentes. Realizam, no caso de comunicações descendentes, a tradução de ordens de fabrico originárias das redes de fábrica, para ordens de execução das tarefas pretendidas, a ser enviadas aos dispositivos do nível inferior. Ou no caso de comunicação ascendente, é responsável pela compilação dos dados recolhidos pelos dispositivos do nível inferior, relativos ao desempenho do sistema ou à evolução das tarefas ordenada, que serão posteriormente enviadas ao nível superior. Neste nível estão normalmente situados os dispositivos de média complexidade, dos quais fazem parte os sistemas SCADA, autómatos programáveis, controladores de robots, etc. Neste nível, o número de equipamentos utilizados, face ao nível anterior e posterior, pode ser considerado um número médio. As mensagens a enviar são de dimensão média/elevada, na ordem dos

Kbytes. Quanto à frequência de transmissão, devem ser de velocidade média/elevada, entre os milissegundos e os segundos. Os seus requisitos temporais podem ser considerados de *soft real-time*, com tempos de respostas inferiores a 100ms. Tal como as redes anteriores, estas funcionam normalmente, para tipos de tráfegos aperiódicos, devem ter uma cobertura geográfica bastante abrangente e uma largura de banda elevada. Ethernet TCP/IP é também, uma solução possível para estes tipos de redes. Uma outra solução possível para este tipo de redes, é a rede CAN-Bus.

5.1.3 - Redes de Campo

As redes de mais baixo nível, são as redes de campo. Estas, utilizam-se para interligar os muitos dispositivos e controladores de campo existentes, como autómatos, microprocessadores, sistemas embebidos, sensores, actuadores, etc. São estas as redes responsáveis pelo funcionamento e controlo do processo de produção de uma fábrica. As mensagens que trocam são de curta dimensão, na ordem dos bytes; o seu tráfego pode ser periódico e/ou aperiódico; as mensagens devem ser trocadas a velocidades muito elevadas; podem ter restrições de tempo real críticas e/ou não críticas (*hard /soft real time*); média ou baixa largura de banda; têm cobertura geográfica limitada; e devem ser redes de baixo custo. Uma solução possível para este tipo de redes é a rede Profibus.

Antes da existência de redes de campo, a ligação entre os dispositivos e controladores de campo era feita a partir de ligações com anéis de corrente 4-20mA e com ligações 0-24V. Este método era muito caro, implicava um uso excessivo de cablagem, que complicava o seu projecto, implementação e manutenção. Com o aparecimento das redes de campo, para além de solucionar estes problemas, criaram-se redes bastante eficazes, que podem mais facilmente ser alteradas, de forma a incluir novos dispositivos na rede.

Uns dos principais problemas deparados para as redes de campo foram os problemas causados pelos ambientes industriais, em que estão inseridas. Estes ambientes, são bastante adversos a comunicações, pois possuem condições atmosféricas muito desfavoráveis, com temperaturas muito altas, grandes percentagens de humidade, muita sujidade, entre outros. Para além de que, nestes ambientes existe, normalmente, um elevado ruído electromagnético devido ao elevado número de máquinas que normalmente existe, o que impõe requisitos adicionais a estas redes, em termos mecânicos e electromagnéticos.

O modelo OSI, apresentado a seguir, serviu de base para a concepção de um modelo para redes de campo, que permitisse representar todos os aspectos relacionados com este tipo de redes. Este novo modelo, é constituído apenas por 3 dessas camadas. São elas: a camada de aplicação; ligação de dados; e física.

A princípio, quando foi criado este modelo, não foi desenvolvida nenhuma norma que especificasse a forma como o modelo devia ser cumprido. O que levou a que fossem desenvolvidas soluções para implementar as funções de cada camada, adaptadas às aplicações para as quais, eram desenvolvidas. A utilização deste método, tornava praticamente impossível a adaptação dos dispositivos, para diferentes fabricantes. Este problema levou, a que no início dos anos 90, se iniciasse um longo processo de criação de um standard para este modelo. Este processo demorou mais de 10 anos e o seu resultado foram as normas internacionais IEC 61158 e IEC 61784 [22].

Resolvida a ausência de uma norma para o modelo de redes de campo, os fabricantes apostaram no desenvolvimento de ferramentas que permitissem um fácil desenvolvimento e

configuração de novas redes, carência que surgiu, do elevado número de dispositivos utilizados hoje em processos industriais. Surgindo assim, nesta área também problemas de padronização dessas ferramentas. Estes problemas podem ser resolvidos, recorrendo ao conceito de *function block*, pela norma IEC 61499.

5.1.4 - Modelo OSI

Num ambiente onde existem equipamentos e redes de comunicações de diferentes fabricantes, a sua integração implica a definição de protocolos de comunicações normalizados. A ISO (*International Organization for Standardization*) definiu o modelo de referência OSI (*Open Systems Interconnection*) com o objectivo de promover o aparecimento de normas na área das comunicações entre computadores.

O modelo OSI é considerado um "sistema de arquitectura aberta", significando que se um sistema estiver conforme este modelo, então está aberto a comunicar com qualquer outro que obedeça às mesmas normas. É de salientar, que o modelo de referência OSI, não especifica por si as normas de comunicação. O seu propósito é apenas fornecer uma arquitectura que sirva de base ao desenvolvimento de normas para sistemas de comunicação [23].

Define assim uma forma possível de comunicação entre sistemas computacionais, dividindo as funcionalidades das redes em 7 camadas: aplicação; apresentação; sessão; transporte; rede; ligação/lógica; e física, como representado na figura 5.1.

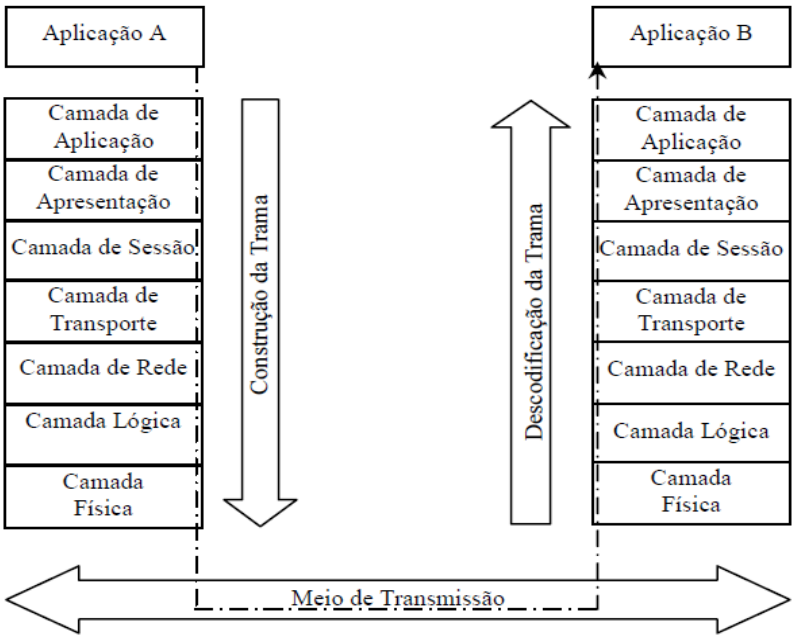


Figura 5.1 - Modelo OSI [23]

O nível inferior do modelo, o nível físico, é responsável pela transmissão dos dados (bits) no meio físico, relativamente às suas características eléctricas ou mecânicas, seja ele um fio de cobre, fibra óptica, ondas de rádio, etc. Controla a velocidade de transmissão desses dados e a quantidade de dados trocados. O objectivo desta camada é "esconder" por completo ao nível de ligação, o tipo de meio físico em que os dados são enviados ou recebidos.

O nível de ligação ou lógica tem como principais funções a sincronização de bits dentro da uma trama, detecção e possivelmente correcção de erros, ocorridos durante a transmissão de informação e por fim tem também como função controlar o fluxo de dados, permitindo assim que dentro de uma mesma rede, possam existir diferentes protocolos de rede. Este camada em certos protocolos pode encontrar-se subdividida em duas outras camada: a LLC (*Logical Link Control*) e a MAC (*Medium Access Control*). Na sub-camada LLC centra-se, principalmente, no controlo dos erros que possam ser verificados nas tramas e na descodificação dos pacotes de dados recebidos, ou na sua multiplexação, caso se pretenda transmitir informação, funcionando assim como interface, para o nível superior de rede. A sub-camada MAC é responsável pelo acesso ao meio, ou seja, pelo acesso à rede, escolhendo por mecanismos específicos normalmente de hardware, qual será do elementos conectados à rede, o que pode transmitir quando vários tentarem acede-lhe ao mesmo tempo, ou quando a rede estiver a ser utilizados e outros elementos quiserem enviar mensagens, impedindo colisões de pacotes; e pelo endereçamento dos elementos da rede, permitindo assim que vários elementos possam partilhar a mesma rede. Esse endereçamento é feito com um endereço MAC, único para cada elemento da rede.

A principal funcionalidade da camada de rede passa, por permitir a comunicação entre dispositivos que não estejam directamente ligados, ou seja, fazer o *routing*/encaminhamento dos pacotes, de maneira a que estes consigam chegar correctamente ao seu destino. Para isso é feito um endereçamento dos pacotes, associando endereços lógicos, como por exemplo endereços IP a endereços físicos, como o endereço MAC.

A camada de transporte tem como principal função, assegurar a qualidade dos serviços de transferência de dados entre aplicações. É uma camada complementar à camada de rede. Podem principalmente proporcionar funcionalidades, tais como: confiabilidade, ao garantir que não há mensagens perdidas ou duplicadas, pelo menos sem que isso seja detectado; ordem, ao entregar as mensagens pela mesma ordem em que são enviadas; e fragmentação/ reconstrução, permitindo trocas de dados mesmo que estes ultrapassem o tamanho máximo dos pacotes, dividindo-os por pacotes de tamanho possível de transmitir. No caso de receber pacotes “partidos”, estes serão aqui recompostos para se obterem os dados a enviar à camada superior.

A camada de sessão, permite que o utilizador e o sistema iniciem o diálogo. A função desta camada passa por estabelecer, manter e gerir esse diálogo [23].

Já a camada de apresentação lida com questões relacionados com o formato dos dados trocados, por exemplo, com o número de bits que os inteiros utilizam, o código utilizado para os caracteres, entre outros, preocupa-se apenas com a forma como é representada a informação, mas não com o significado desta mesma informação.

O nível de aplicação, refere-se a uma aplicação propriamente dita, permite ao utilizador trocar mensagens com o sistema de comunicação global, sem ter que se preocupar com questões tratadas pelas camadas inferiores, como protocolos utilizados, erros ocorridos, etc.

Na internet, é utilizado um modelo mais simples, chamado *Internet Model*, construído a partir do modelo OSI, mas apenas com 5 camadas: física, ligação, rede, transporte e aplicação, tendo esta última, incluídas as funcionalidades, tanto da camada de sessão, como de apresentação, como de aplicação do modelo OSI.

5.2 - Rede CAN-Bus

5.2.1 - Especificação CAN

A *Controller Area Network* (CAN) é um protocolo de comunicação série que pode ser aplicada em controlo distribuído de tempo real, com um nível de segurança muito elevado [24].

Este protocolo, começou a ser desenvolvido em 1983 por Robert Bosh GmbH e foi aprovado em 1986 no congresso da SAE (*Society Automotive Engineers*). A sua área de aplicação seria essencialmente para a indústria automóvel, mas que hoje em dia é utilizado também noutras áreas, como em redes de comunicação industriais. Os primeiros microprocessadores CAN apareceram em 1987, fabricados pela Intel e pela Philips. Hoje praticamente todas as marcas de microprocessadores apresentam soluções com esta tecnologia embutida.

A Bosh publicou a 1991 a especificação do protocolo até hoje utilizada, *CAN specification 2.0*.

Entre outras vantagens, a rede tem um custo reduzido, manutenção fácil e pouca complexidade de implementação. Tendo como principais características [25]:

- Estrutura multi-mestre com acesso múltiplo;
- Velocidade de transmissão máxima de 1Mbps;
- Meio físico idêntico ao do RS-485, transmissão diferencial, com cabo entrançado, preferencialmente blindado;
- Todas as mensagens têm um nível de prioridade;
- Controlo de erro totalmente implementado com circuitos integrados dedicados;
- Mensagens de dados com campo de dados de comprimento entre 0 e 8 bytes;
- Grande segurança de transmissão e grande fiabilidade;
- Baixo custo;
- Capacidade de funcionamento em ambientes hostis;
- Simples de implementar a interface entre o barramento CAN e o CPU de circuitos integrados dedicados (controladores de barramento CAN).

A rede CAN (ou CAN-Bus) pode ser considerado um barramento série, que apresenta características multi-mestre, isto é, todos os nós da rede têm a capacidade de iniciar uma transmissão de dados por iniciativa própria, o que pode originar que todos os nós/mestres presente no barramento tentem iniciar a comunicação em simultâneo. Uma outra característica interessante, é a capacidade de transmissão de dados em tempo real, no entanto, a quantidade de dados enviados, de cada vez, deve ser pequena.

A especificação CAN inclui exclusivamente, as camadas físicas e de ligação de dados do modelo OSI [21], descrito anteriormente. No entanto, foram já desenvolvidas, uma enorme variedade de *standards*, que implementam as camadas superiores do Modelo OSI, particularmente e camada de aplicação, das quais se destacam o CANOpen e o Device Net.

Camada Física

Ao nível da camada física os dados são enviados por apenas dois fios, o CANH (*CAN High*) e o CANL (*CAN Low*) de forma diferencial, o que torna esta comunicação muito robusta e imune ao ruído, pois a inclusão de ruído numa das linhas, implica a inclusão de ruído na outra,

continuando assim a diferença de tensões nas linhas igual. A transmissão dos bits é realizada então segundo a figura seguinte (figura 5.2).

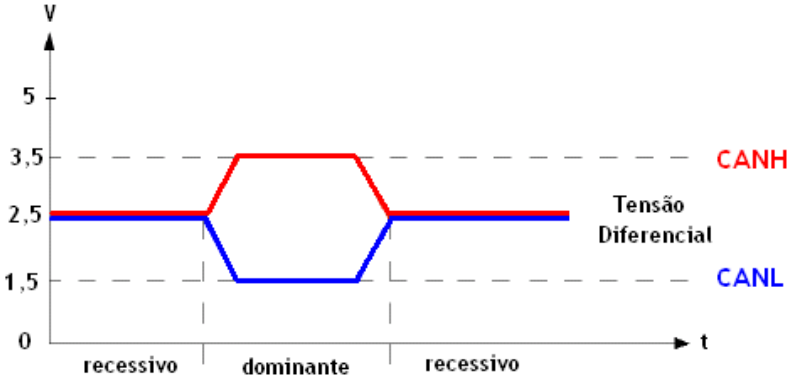


Figura 5.2 - Transmissão de dados CAN no nível físico

Um sinal recessivo equivale digitalmente ao sinal 1, um sinal dominante corresponde ao um 0 na forma digital.

Uma outra vantagem da transmissão diferencial é permitir a transmissão em longa distância sem necessidade de repetidores, no entanto essa distância depende entre vários factores da velocidade de transmissão utilizada. Na tabela 5.1, pode ser verificada a relação entre a distância, velocidade, tipos de cabos, resistências dos cabos e resistências a utilizar como terminadores.

Tabela 5.1 — Relação entre velocidade de transmissão e características técnicas dos cabos CAN

Velocidade	Tamanho do Barramento	Tipo de cabo / Diâmetro	Resistência do cabo	Terminador
50 kbit/s	Máximo 1000 m	AWG18 0.75 - 0.8 mm^2	70 mΩ	150 - 300 Ω
125 kbit/s	Máximo 500 m	AWG20 0.5 - 0.6 mm^2	< 60 mΩ	120 - 300 Ω
500 kbit/s	Máximo 300 m	AWG22, AWG20 0.34 - 0.6 mm^2	< 40 mΩ	120 Ω
1000 kbit/s	Máximo 40 m	AWG23, AWG22 0.25 - 0.34 mm^2	< 26 mΩ	120 Ω

Os valores em cima apresentados, não são únicos, sugerem apenas uma relação entre a velocidade e o tamanho do barramento. O tamanho do barramento pode ser aumentado se no seu final for incluído um repetidor que permite regenerar o sinal transmitido, dobrando o tamanho máximo admissível para um barramento, então se pretendermos um barramento de 10Km que troca dados a uma velocidade de 50kbps, teriam de ser utilizados 10 cabos de 1km e 9 repetidores. Os terminadores de uma rede não são mais que resistências entre o CANH e o CANL, que permitem uma perfeita propagação dos sinais na rede, garantido a reflexão do sinal transmitido.

Em baixo (figura 5.3) pode ser observado um exemplo de um barramento CAN com três dispositivos conectados.

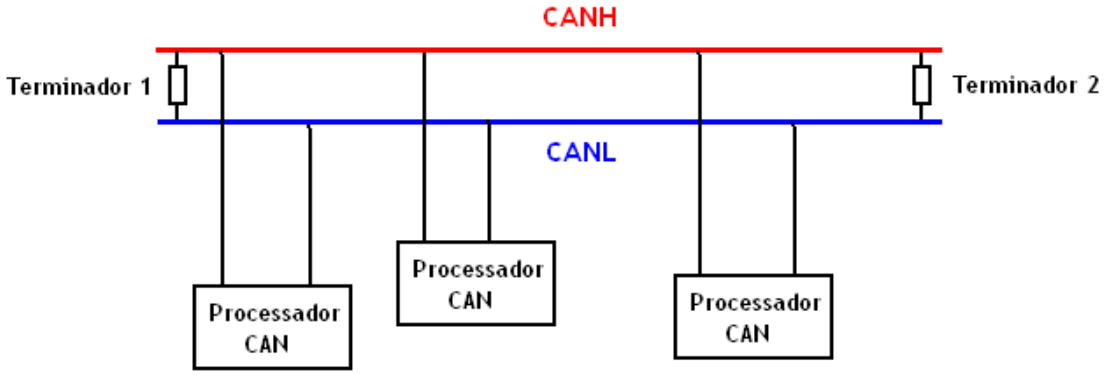


Figura 5.3 - Exemplo de Barramento CAN

O protocolo CAN por permitir velocidades de transmissão tão variadas, está definido em duas normas internacionais de comunicações, a ISO 11898 para alta velocidade e a ISO 11519-2 para baixa velocidade. Para que a rede pudesse estar de acordo com a ISSO 11898 a sua velocidade de transmissão deve estar entre os 125kbps e os 1Mbps, o que implica que o barramento tenha uma dimensão máxima de 500 m. Para acordar com a norma ISO 11519-2 a sua velocidade não pode ultrapassar os 125kbps.

Tempo de Transmissão de um bit

O tempo para transmissão de um bit, está dividido em quatro segmentos de tempo: segmento de sincronização (SYNC_SEG); segmento de tempo de propagação (PROP_SEG); segmento de fase 1 (PHASE_SEG1) e segmento de fase 2 (PHASE_SEG2), como representado na figura 5.4. Cada segmento é constituído por um número inteiro de *time quantums* (TQ), que representam a mais pequena unidade de tempo imposta pela rede CAN.

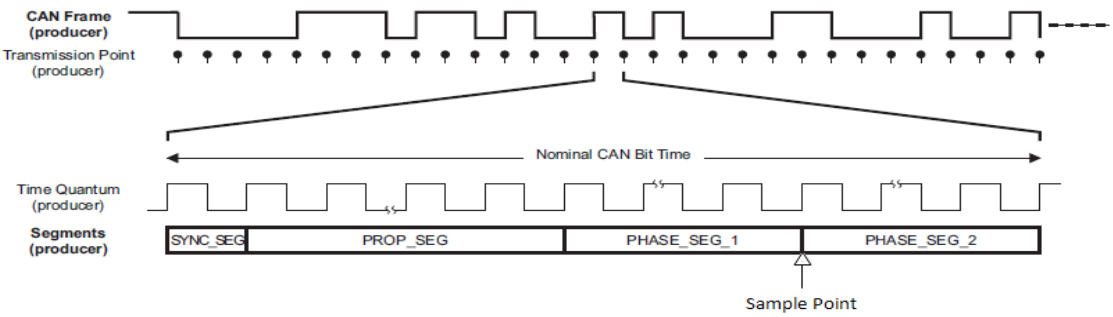


Figura 5.4 - Construção temporal de um bit CAN (adaptado de [26])

Durante o primeiro segmento de tempo do envio de um bit, segmento de sincronização, é realizada, tal como o nome indica, a sincronização de todos os nós do barramento. É também neste segmento que dever ser realizada a mudança de estado de estado da rede, de dominante para recessivo ou vice-versa, se houver essa necessidade.

O segmento de tempo de propagação, permite compensar os atrasos da transmissão dos sinais na rede. Estes atrasos, podem ser causados quer pela transmissão dos sinais no próprio barramento, quer pelos atrasados provocados na passagem dos sinais nos *transceivers*.

Os segmentos de fase 1 e 2 são necessários para compensação de possíveis erros de fase da transmissão dos sinais nos cabos CAN. Estes segmentos podem ser alongado ou encurtados durante a sincronização [24].

Entre estes dois últimos segmentos, encontra-se o ponto de amostragem, que é relativo ao instante de tempo em que o nível da rede é lido e interpretado, como um valor lógico de bit (1 ou 0), por todos os nós do barramento.

Camada de Ligação de Dados

Ao nível da ligação, o barramento CAN apresenta mecanismo de detecção e tratamento de erros e mecanismos de controlo do acesso ao meio, que são suportados pela estrutura das tramas definidas neste protocolo.

O modelo de cooperação, utilizado entre os dispositivos conectados no barramento, é do tipo Produtor – Consumidor. Utilizam-se mecanismos de transmissão *broadcast*, em que uma mensagem envia por um dos nós é recebida por todos os outros, que devem possuir mecanismos que permitam seleccionar apenas a informação que pretendem, estes mecanismos realizam esta selecção normalmente a partir de um identificador da mensagem enviada. Este identificador da mensagem deve ser único em toda a rede, e define não só o conteúdo, mas também a sua prioridade.

As prioridades de cada mensagem são previstas durante o design do sistema, a partir dos seus valores binários e não podem ser alterados dinamicamente. O identificador com o menor valor binário é o que tem maior prioridade.

Existem quatro tipos de tramas definidas neste protocolo:

- Trama de dado, utilizada para transmissão dos dados;
- Trama remota, que é utilizada para que um nó em modo receptor, peça a outro nó da rede que lhe envie informação;
- Trama de Erro, que indica que uma trama enviada para o meio por um nó, foi visualizada com erros por outro nó;
- A Trama de Sobrecarga, é enviada quando o tempo para os nós se preparar para receber uma trama não foi suficiente.

Tramas CAN

Segundo a especificação do protocolo as tramas de dados podem ser de dois formatos diferentes o formato *standard* que possui um identificador de 11 bits (*CAN Specification 2.0 Part A*), ou o formato estendido em que as tramas apresentam identificadores de 29 bits (*CAN Specification 2.0 Part B*). Estas duas tramas estão representadas na figura 5.5.

O primeiro campo é constituído pelo bit SOF (*Start of Frame*), um bit dominante que indica a intenção de um dispositivo de iniciar a transmissão de uma mensagem.

Seguidamente vem o campo de arbitragem, que engloba o identificador da mensagem e *Remote Transmission Request* (RTR), bit indicador de trama remota se for recessivo, ou trama de dados se for dominante. No caso de se pretender uma mensagem com identificar de 29 bits, o RTR e o bit seguinte o IDE (*Identifier Extension*) vêm como recessivos sendo que na

posição inicial do RTR passa a estar o bit SRR (*Substitute Remote Request*) e o RTR passa para o final da extensão do identificador.

Posterior ao campo de arbitragem vem o campo de controlo que incluí o DLC (*Data Length Code*), onde é definida a quantidade de bytes de dados que contém a trama, sendo o valor máximo de oito bytes.

À frente do DLC aparecem os dados propriamente ditos, seguidos do CRC que permitem verificar a integridade dos dados da mensagem, tema abordado na secção 2.1.6 de segurança em sistemas de automação.

O campo de reconhecimento (*Acknowledge*) que aparece depois dos dados é composto por 2 bits: o *ACK slot* e o *ACK delimiter*. O *ACK slot* é enviado como bit recessivo e é colocado a dominante pelos dispositivos receptores que leram a mensagem correctamente.

Por fim, aparece um conjunto de 7 bits em modo recessivo que indicam o fim da mensagem (*End of Frame*).

Uma nova mensagem só poderá ser enviada quando o tempo de intervalo (*Intermission Frame Space*) de 3 bits for ultrapassado.

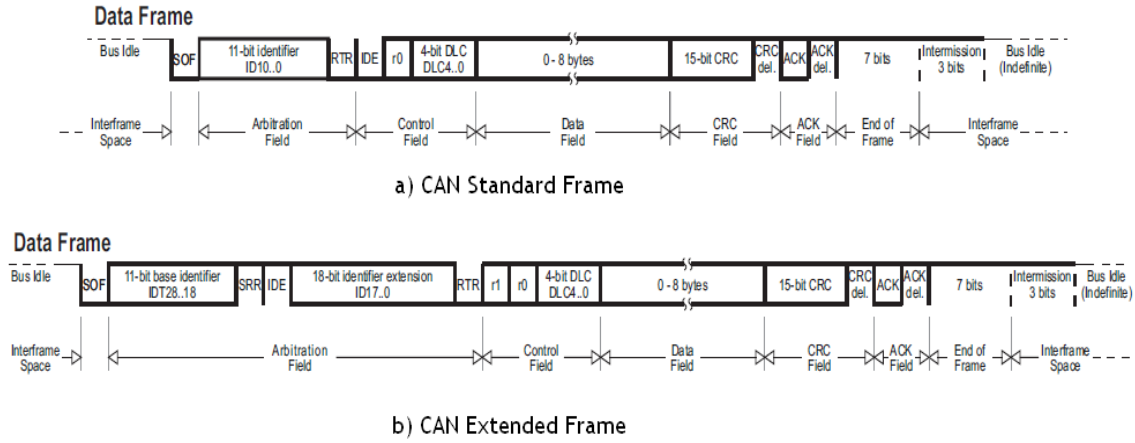


Figura 5.5 - Tramas do Protocolo CAN-Bus: a) Trama standard e b) Trama estendida [26]

Arbitragem

Quando dois ou mais dispositivos ligados à rede tentam aceder-lhe é realizado uma arbitragem não destrutiva, que permite o acesso ao meio, ao dispositivo com a mensagem de maior prioridade.

Essa arbitragem é baseada num *bit-wise arbitration*. Então um nó que pretenda transmitir uma mensagem envia um SOF e neste momento todos os nós sincronizam-se com este. Se algum dos nós pretender também nesse instante transmitir uma mensagem envia o seu identificador ao mesmo tempo que o nó que tomou a iniciativa também o envia. A partir daqui a arbitragem é realizada de acordo com o mecanismo "*Wired-And*", em que os estados dominantes se sobrepõem aos estados recessivos. Afirmando-se assim que CAN utiliza CSMA/CD (*Carrier Sense Multiple Access / Collision Detection*) para arbitrar o acesso ao meio dos dispositivos num barramento. Este mecanismo de arbitragem está representado na figura 5.6.

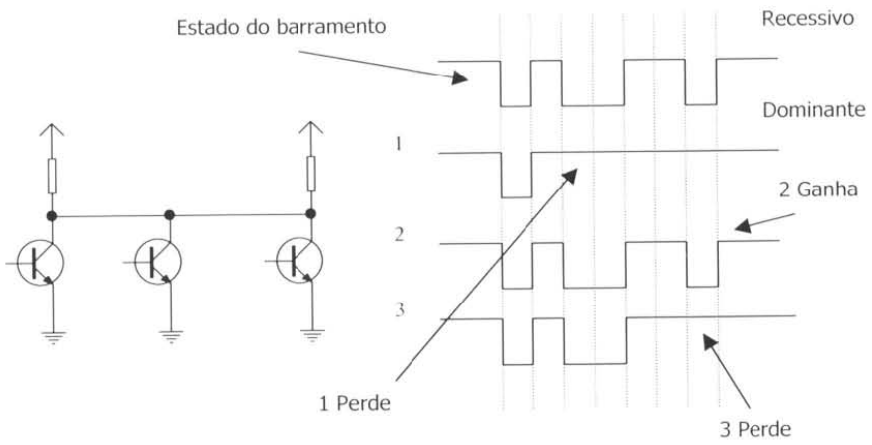


Figura 5.6 - Mecanismo de arbitragem do acesso ao meio na rede CAN-Bus [25]

Na figura anterior pode ver-se à esquerda o circuito responsável pelo mecanismo de "*Wired-And*", sendo que, a cada um dos transístores estaria associado um dispositivo que pretende aceder ao meio. Este circuito permite que seja realizada entre os sinais de dados a operação lógica de *AND*, que produz um sinal dominante (lógico 0) quando à sua entrada observa um sinal recessivo (lógico 1) e um sinal dominante. Assim vence a arbitragem o dispositivo que apresentar no identificador da sua mensagem, o menor valor binário.

Os dispositivos perdedores, tornam-se automaticamente receptores e só voltam a tentar transmitir quando o barramento estiver livre.

Detecção de Erros

O protocolo CAN assinala um erro imediatamente após a sua ocorrência. Apresenta três mecanismos de detecção de erros ao nível da mensagem e dois ao nível dos bits [26]. Sempre que um dispositivo verifica um erro, envia para o barramento uma trama de erro e o transmissor tenta novamente enviar a mensagem.

Ao nível da mensagem realizam-se os seguintes métodos de detecção de erros:

- Método do CRC;
- Método de verificação da trama (*Frame Check*), em que o dispositivo verifica bit a bit se a estrutura da trama recebida está de acordo com o formato e tamanho especificados;
- Erro de *Acknowledge*, ocorre quando o transmissor não recebe o sinal de *Acknowledge* por parte do receptor.

Ao nível dos bits os métodos utilizados são:

- Método de monitorização - os nós transmissores quando estão a enviar uma mensagem ao mesmo tempo, estão a escutar a linha e se o bit enviado for diferente do bit recebido, significa a ocorrência de um erro, que pode ser um erro local do transmissor;
- E o método do Bit Stufffing - em que após cinco bits iguais consecutivos, o emissor insere um sexto bit de valor oposto aos anteriores, *o stuff bit*, que será aquando da leitura removido pelos receptores. Se algum dispositivo detectar seis bits seguidos com valor igual, sinaliza automaticamente a ocorrência de um erro.

Gestão e Limitação de Erros

Um nó de uma rede CAN-Bus pode encontrar-se num dos seguintes estados:

- Erro Activo – neste estado o nó é um membro activo da rede e sempre que detecta um erro, coloca na rede uma trama de erro activo;
- Erro Passivo – neste estado o nó continua a ser um membro activo da rede, no entanto quando detecta um erro, este nó, em vez de enviar uma trama de erro activo, envia para a rede uma trama de erro passivo e após a sua transmissão este espera até que uma outra estação inicie uma nova transmissão de dados;
- *Bus Off* - quando um nó se encontra neste estado, significa que esse nó não se encontra conectado à rede, não tendo por isso nenhuma influência sobre ela.

Cada nó conectado ao barramento CAN possui contadores de erros de transmissão (TEC) e contadores de erro de recepção (REC). Estes contadores são incrementados sempre que acontece um erro na operação respectiva e são decrementados sempre que essa mesma operação ocorreu correctamente. A máquina de estados relativa à gestão dos erros por parte dos nós CAN, esta representada na seguinte ilustração (figura 5.7).

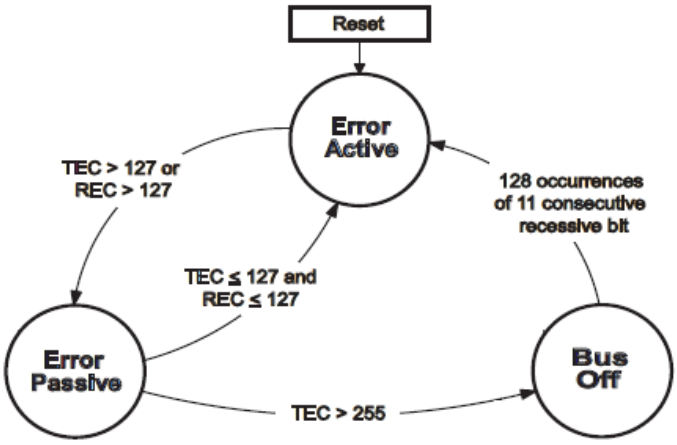


Figura 5.7 - Máquina de estados de gestão dos erros dos nós de um barramento CAN (adaptado de [26])

Camada de Aplicação

A norma CAN, tem uma das maiores famílias de *standards* nos níveis OSI mais elevados.

Um dos *standards* mais conhecidos é o CANOpen, que é baseado no *standard* DS 301 da CiA (*CAN in Automation*). É muito utilizado em sistemas embebidos e permite a criação de uma rede muito dinâmica para dispositivos CAN. Tem um mecanismo de transmissão de dados orientado a objectos e apresenta como subprotocolos mais comuns: o PDO (*Process Data Object*), o SDO (*Service Data Object*), o NMT (*Network Management*), o SYNC (*Synchronization Object*) e o EMG (*Emergency Object*).

Outro é o *CAN Aerospace*, desenvolvido pela NASA (*National Aeronautic and Space Administration*) e é utilizado para controlo de navegação dos sistemas.

O Device Net, criado pela empresa *Allen-Bardley - Rocwell* e aprovado pela CiA, é um *standard* mais largamente utilizados em aplicações de industrias de automação. Permite a comunicação entre os dispositivos de entradas e saídas e os seus controladores [27].

O *Smart Distributed System* utilizado em sistemas de medição e controlo. É um *standard* aprovado pela CiA e criado pela *Honeywell*.

Existem ainda, normas utilizadas para sistemas de transportes, como a *Safety Bus*. Normas utilizadas para a comunicação componentes de veículos, como a SAE J1939, criada pelo *Society of Automotive Engineers*. Para aplicações agrícolas, existem a ISO 11783 ou a DIN 9684-LBS. Para aplicações navais e aéreas, como por exemplo a NMEA 2000, entre muitas outras.

5.2.2 - CAN-Bus no AT90CAN64

O microprocessador AT90CAN64, possui um controlador de rede CAN completamente compatível com a especificação CAN 2.0 parte A e parte B [26]. Implementando as camadas física e de ligação do Modelo OSI e permitindo larguras de banda até 1Mbits/s.

Neste controlador, é encontrado todo o hardware necessário para o envio, recepção, filtração e gestão das mensagens de uma rede CAN.

Para que uma mensagem a ser enviada ou recebida, este módulo contém quinze estruturas, chamadas *Message Object* (MOb). Estes MOb, podem ser vistos como descritores das tramas CAN, que contêm a informação relativa a esse trama. Nestes objectos, deve então ser definido: o modo de operação da mensagem, isto é, se a mensagem relativa ao MOb é uma mensagem para transmitir, ou para receber e neste último caso se é ou não uma mensagem mascarada (conceito explicado mais à frente); qual o identificador da mensagem, bem como se ela é uma mensagem *standard* ou estendida e ainda, o número de bytes de dados que a trama contém (no máximo 8). Os objectos MObs, possuem ainda outras grandezas, como *timestamps* e outras variáveis de controlo, que não são directamente programadas, são utilizadas apenas pelo microprocessador para tarefas de controlo do fluxo de dados da rede.

As mensagens configuradas nos MObs, como mensagens a receber, podem ser: mensagens totalmente mascaradas; parcialmente mascaradas, ou não mascaradas. Este conceito está relacionado com um filtro de mensagens programável, existentes nestes controladores CAN, do AT90CAN64, que permite a aceitação parcial ou completa das mensagens da rede.

Assim, um MOb referente a uma mensagem de recepção totalmente mascarado, apenas lê a mensagem colocada na rede, se esta usar um identificador coincidente com o identificador esperado para o MOb. Uma mensagem de recepção parcialmente mascarada é uma mensagem na qual alguns dos bits do seu identificador (número definido pelo programador), têm que coincidir com os bits, nas mesmas posições, definidas previamente. Por último, nas mensagens de recepção não mascaradas, são recebidas todas as mensagens colocadas na rede, independentemente do seu identificador.

5.2.3 - Protocolo CAN

Como já foi dito anteriormente, será utilizada uma rede CAN-Bus como rede de controlo para o sistema. Como as tramas CAN apenas permitem 8 bytes de dados, não é muito conveniente utilizar o protocolo apresentado em na secção 3.2.3, porque uma mesma mensagem poderia implicar a utilização de várias tramas CAN para enviar todos os dados, particularmente para as mensagens de definição da função de leitura dos Escravos. No caso do protocolo geral, é ainda utilizado um byte por caracter, neste caso parece mais razoável utilizar um único byte para enviar símbolos com valor ASCII inferior a 256, por exemplo, o

valor 123 pode ser enviado num só byte, em vez de ser enviado em três, contendo respectivamente os caracteres 1, 2 e 3. Foi então definido um novo protocolo, bastante similar ao protocolo geral, mas mais reduzido, aproveitando melhor os bytes das tramas. Esse protocolo será apresentado em seguida.

Protocolo CAN

Uma grande diferença deste novo protocolo, em relação ao protocolo geral, é a exclusão do ID do Mestre, esta remoção foi realizada, devido à necessidade de diminuir o número de tramas CAN utilizadas para enviar uma mesma mensagem. Como todos os Escravos possuem o seu próprio ID, não há necessidade ao nível das comunicações de incluir este identificador do Mestre, porque pelo ID do Escravo facilmente se obtém o ID do Mestre. Esta simplificação, pode funcionar em qualquer sistema que possua no máximo 255 Escravos, com ID diferentes, que é o valor já bastante assinalável, pois se considerarmos que cada um dos Escravos terá 7 sensores (número de ADCs dos Waspmotes) teríamos uma estrutura com 7 x 255 = 1785 sensores, mais os sensores que poderiam ainda ser introduzidos nos Mestres. Não será portanto muito provável, mas poderão existir estruturas que necessitem de mais de 255 Escravos e neste caso, seria necessária uma revisão desta abordagem.

Uma outra diferença do protocolo CAN em relação ao anterior diz respeito ao CRC. Como já foi referido na introdução teórica deste capítulo, os dispositivos de uma rede CAN-Bus, implementam os seus próprios mecanismos de detecção de erros, sendo que um deles é CRC.

Os restantes dados do protocolo CAN e no protocolo geral são praticamente os mesmos.

Tramas de Alarmes - Tipo W

Para estas mensagens, é necessário apenas uma trama CAN com todos os seus dados (figura 5.8). Os dados utilizados nesta trama, são exactamente os mesmos que os da trama equivalente do protocolo geral, no entanto como o parâmetro pode apresentar um valor superior a 1 byte, este é enviado nesta trama em 2 bytes: Valor_1, com a parte mais significativa e Valor_2 com a parte menos significativa do valor respectivo.

Bytes [7:0]							
0	1	2	3	4	5	6	7
ID_Escravo	ID_Mensagem	Tipo_Trama	Tipo_Alarme / Pedido	Descrição	Valor_1	Valor_2	-

Figura 5.8 - Dados da Trama de Alarmes do protocolo CAN

Trama de Sincronização dos Relógios – Tipo C

Para esta mensagem, são já necessárias duas tramas CAN. Os dados aqui presentes são também os mesmos que para a mensagem original. Neste caso, é necessário transmitir também em dois bytes o ano, por ser um valor superior a 255. Os dados são então enviados, conforme a figura 5.9.

	Bytes [0:7]							
	0	1	2	3	4	5	6	7
C.1	ID_Global	ID_Mensagem	Tipo_Trama	Ano_1	Ano_2	Mês	Dia	Hora
C.2	Minutos	Segundos	-	-	-	-	-	-

Figura 5.9 - Dados da Trama de Sincronização dos Relógios do protocolo CAN

Trama de Definição das Funções dos Escravos – Tipo F

A trama de definição das funções dos escravos é a que apresenta mais diferenças em relação à trama original, essencialmente no parâmetro da função a definir no Escravo. É necessário neste caso, a utilização de 3 tramas CAN, seguidas de 8 bits. Como se pode verificar na figura 5.10.

	Bytes [0:7]							
	0	1	2	3	4	5	6	7
F.1	ID_Escravo	ID_Mensagem	Tipo_Trama	FunçãoP	FunçãoS	Ano_Ini_1	Ano_Ini_2	Mês_Ini
F.2	Dia_Ini	Ano_Fim_1	Ano_Fim_2	Mês_Fim	Dia_Fim	Hora_Ini	Minut_Ini	Sec_Ini
F.3	Hora_Fim	Minutos_Fim	Segundos_Fim	Num_Sensores	ID_Sensor1	ID_Sensor2	ID_Sensor3	ID_Sensor4

Figura 5.10 - Dados da Trama de Definição das Funções dos Escravos do protocolo CAN

Na primeira trama CAN desta mensagem, estão presentes, para além dos três bytes do cabeçalho comuns, dois bytes relativos à função a que se pretende definir os Escravos. No primeiro destes bytes, é definida a função principal, ler continuamente, ler num intervalo de tempo, ou adormecer o Escavo (*sleep mode*), utilizando as letras C, I e S respectivamente. No segundo, indica-se nos casos em que se aplique, se a função é definida a todos os sensores, ou apenas para um número de sensores restrito, que será indicado a seguir com as letras A e S respectivamente.

Os restantes dados da trama, são exactamente os mesmos que os da trama original, apenas se deve realçar, que para envio do ano quer iniciais, quer finais são necessários dois bytes.

Trama de Leitura dos Sensores dos Escravos – Tipo R

Neste caso particular é utilizada apenas uma trama CAN para realizar os pedidos e são utilizadas duas tramas CAN para a resposta do Escravos. Os dados utilizados são os mesmos que os do protocolo original. A definição destes dados está ilustrada na figura 5.11. Apenas há a realçar neste caso, que os resultados das medições dos sensores, estão representados também em dois bytes.

	Bytes [0:7]							
	0	1	2	3	4	5	6	7
Pedido - R	ID_Escravo	ID_Mensagem	Tipo_Trama	-	-	-	-	-
Resposta - R.1	ID_Escravo	ID_Mensagem	Tipo_Trama	Num_Sensores	ID_Sensor1	Result_Sens1_1	Result_Sens1_2	ID_Sensor2
Resposta - R.2	Result_Sens2_1	Result_Sens2_2	ID_Sensor3	Result_Sens3_1	Result_Sens3_2	ID_Sensor4	Result_Sens4_1	Result_Sens4_2

Figura 5.11 - Dados da trama de leitura dos sensores dos Escravos

Trama de Leitura de Vários Valores dos Sensores dos Escravos – Tipo L

As tramas de "controlo" deste tipo de mensagens são idênticas às do protocolo original, como pode ser observado na figura 5.12.

Bytes [0:7]							
0	1	2	3	4	5	6	7
ID_Escravo	ID_Mensagem	Tipo_Trama	ID_Sensor	Índice_Mensagem	Num_Tramas	Num_Dados	-

Figura 5.12 - Dados das tramas de leitura de vários valores dos sensores dos Escravos

As tramas CAN que contêm os dados medidos pelos sensores, apresentam apenas 8 resultados em cada uma. Isto deve-se há necessidade de que esses valores sejam enviados em dois bytes, pois podem apresentar resultados com valores superiores a 255. Então, na rede CAN para serem enviados os 20 dados de uma trama completa do protocolo geral, são necessários 20/4 = 5 tramas.

Trama de Leitura e Mudança de Estado dos Actuadores dos Escravos – Tipo A e M

São também idênticas às mensagens do protocolo original. É necessária apenas uma trama CAN para o envio destes dados (figura 5.13).

Bytes [0:7]							
0	1	2	3	4	5	6	7
ID_Escravo	ID_Mensagem	Tipo_Trama	ID_Actuador	Estado	-	-	-

Figura 5.13 - Dados das tramas de leitura e mudança de estados dos Actuadores dos Escravos

5.2.4 - Implementação das Gateways CAN – RS-232

Configurações da Rede CAN no AT90CAN64

A configuração da rede CAN-Bus no AT90CAN64, implica a manipulação de uma quantidade bastante elevada de registos deste processador, relativos ao modo de comunicação, à velocidade de transmissão, aos MObs, entre outros. São um total de 17 registos para a configuração da rede CAN em geral, mais 6 registos para configurações dos *Message Objects*. A configuração de todos estes registos, previa-se portanto um processo muito moroso. Processo este, que foi bastante simplificado graças a uma biblioteca disponibilizada pelos fabricantes dos microprocessadores, a Atmel, que permite de forma mais simplificada, a configuração de toda a rede, particularmente para a velocidade de transmissão de dados na rede e dos MObs.

Na secção relativa à arquitectura do sistema, foi realçado que será utilizada uma velocidade de comunicação de 100 kbps na rede CAN-Bus, que nos permite atingir com um único cabo, distâncias acima dos 500 m (visto que para 500 m o máximo de velocidade possível são 125 kbps). Será portanto, utilizada a rede CAN segundo o protocolo de baixa velocidade ISO 11519-2, que integra redes de comunicação com velocidades até 125 kbps. Na biblioteca disponibilizada, é possível definir directamente numa variável a velocidade de transmissão pretendida.

No que diz respeito aos MObs há quatro configurações prévias, necessárias a fazer: o modo de funcionamento (comando) da mensagem dos MObs (transmissão, recepção sem máscara, ou recepção mascarada); o identificador da cada mensagem, que nesta aplicação será apensas *standard*; o número de bytes da mensagem de dados (DLC) e por fim, o vector de dados pertencente ao MOb, que será alterado de acordo com a mensagem a enviar.

Nesta fase, parece relevante realizar uma pequena explicação relativa a dois conceitos distintos, verificados na biblioteca CAN relativa aos MObs, são eles: inicialização e criação do MOb. Como foi referido anteriormente, o AT90CAN64 apenas suporta 15 MObs, mas esta limitação é relativa apenas aos 15 MObs criados, não a MObs (variáveis do programa) inicializados. A inicialização diz respeito à atribuição, na memória do programa das configurações acima enumeradas. Não há um limite, a não ser o da memória do microprocessador, para o número de "MObs" (variáveis do programa) inicializados. Esta inicialização é realizada ao nível do código do programa apenas. Não sendo esses valores, atribuídos directamente, durante a inicialização, a nenhum MOb da página dos 15 MObs do microprocessador, essa tarefa é realizada na sua criação.

Então, a criação de um Mobs refere-se a duas tarefas do microprocessador: a inclusão de um MOb, previamente inicializado, a um MOb (real) da página dos 15 MObs do microprocessador; e ainda, à execução por parte do microprocessador, do seu comando pretendido (de transmissão ou recepção). Ou seja, quando um MOb é criado, para além de ser incluído numa das memórias dos 15 MObs reais, é ainda no caso de ser um MOb de transmissão incluído na rede, ou no caso de ser um MOb de recepção escutado da rede. Esta função de criação, retorna portanto, ao fim de um curto espaço de tempo, uma confirmação ou uma falha na execução da tarefa de criação (transmissão ou recepção). Quando este retorno (confirmação ou falha) se dá, o MOb inicializado, da página de Mobs do microprocessador é automaticamente libertado, para poder ser ocupado por outro MOb. Este processo de criação, realiza-se assim, apenas nos instantes em que se pretende enviar ou receber uma mensagem. Pode por isso, designar-se este processo por criação e comunicação dos MObs.

Fica então assim evidente, que o número de 15 MObs, diz respeito ao número de mensagens, que o microprocessador pode estar à espera de enviar (rede disponível) ou receber ao mesmo tempo.

Inicialização dos MObs

Nesta aplicação, será utilizado apenas a especificação CAN 2.0 parte A, cujas mensagens tem um identificador de 11 bits, permitindo utilizar no sistema 2048 mensagens distintas, que é um número mais que suficiente, para implementar o protocolo CAN definido, para um sistema deste tipo, como ficará evidente mais à frente. A utilização de mensagens estendidas (CAN 2.0 B), por utilizar 29 bits no seu identificador, permite que no sistema sejam utilizadas 536870912 mensagens distintas. No entanto, esta especificação torna a rede mais lenta, por ser necessário enviar mais bits por cada trama de dados. É apenas aconselhado utilizar esta especificação em sistemas que necessitem de mais que 2048 mensagens com identificadores diferentes.

Como foi citado na apresentação teórica da rede CAN (secção 5.2), o identificador das mensagens é o campo do cabeçalho das tramas CAN, que lhes fornece prioridade de envio na rede. Quanto menor o valor binário do identificador, mais prioridade na rede terá a mensagem. Este facto, permite que sejam atribuídas prioridades às mensagens de maior

importância para um sistema. Pegando então nesta característica, atribuíram-se as seguintes prioridades descendentes às tramas especificadas no protocolo CAN: trama de alarmes (trama W); trama de sincronização dos RTCs (trama C); definir a função de leitura dos Escravos (trama F); ler as medições dos sensores dos Escravos (trama R); ler o estado dos actuadores (trama A); mudar o estado dos actuadores (trama M); e finalmente leitura de muitos valores dos sensores (trama L).

Nos programas para as *gateways* desenvolvidos, foi inicializado um MOb por cada mensagem possível no sistema, ou seja, por exemplo para as tramas de leitura dos valores dos sensores, foram inicializados dois MObs de pedidos, um de pedidos do Mestre Central ao Mestre 1 e outro de pedidos do Mestre Central ao Mestre 10 e ainda mais quatros MObs de resposta: um para a parte 1 da resposta do Mestre 1 ao Mestre Central e outro para a parte 2 e outros dois idênticos, da resposta do Mestre 10 ao Mestre Central. Portanto, o total de MObs inicializados foi de 45: 4 para tramas W; 8 para as tramas C; 12 para as tramas F; 6 para as tramas R; 4 para as tramas A; 4 para as tramas M; 6 para as tramas L; e finalmente, foi inicializado um outro MOb, designado de "*CAN_message*", utilizado para recepção de tramas de modo não mascarado, cuja utilização será explicada mais à frente.

Imagine-se então, que para cada mensagem eram necessárias 6 tramas CAN, como no pior dos casos anteriores (tramas F), por cada Mestre existente no sistema, ou seja seriam 6x7 = 42 tramas CAN por Mestre, sendo 7, o número dos diferentes tipos de tramas existentes no protocolo CAN. O máximo de Mestres que poderiam existir na rede utilizando a especificação CAN 2.0A seria de 2048/42 ≈ 48. Considerando que existe um Mestre por cada pilar da ponte, a ponte monitorizada utilizando esta especificação, pode ter no máximo 48 pilares, que é um número muito significativo.

As prioridades das mensagens destes MObs, foram atribuídas de modo geral da forma especificada em cima. Foi, no entanto, atribuída maior prioridade às mensagens de pedidos do Mestre Central, relativamente às mensagens de resposta dos Mestres e foi ainda aleatoriamente, atribuída maior prioridade às mensagens enviadas e recebidas do Mestre 1, em relação às do Mestre 10.

O seguinte parâmetro, necessário de inicializar nestas mensagens, é o seu modo de funcionamento. As mensagens de pedidos do Mestre Central aos Mestres, foram inicializadas no Mestre Central como mensagens a transmitir e nos Mestres inicializadas como mensagens a receber. Para as mensagens de resposta dos Mestres, para o Mestre Central, a situação inverte-se. Ainda neste parâmetro, foram como recepções não mascaradas todas as primeiras (ou únicas) tramas, de uma mensagem do protocolo CAN. Ou seja, para a mensagem F, por exemplo, definiu-se que a primeira parte da mensagem (a primeira trama CAN) é não mascarada e a segunda e terceiras partes são mascaradas. Esta opção, realiza-se para que os dispositivos quando estão à escuta da rede aceitarem todas as mensagens que são nela colocadas, fazendo posteriormente a selecção dessa mensagem pelo ID do Escravo. Este processo será explicado mais à frente.

Outro parâmetro inicializado nas tramas CAN, é o número de bytes de dados utilizados na mensagem. É necessário definir para a primeira (ou única) mensagem CAN um valor de 8 no DLC, mesmo que não seja necessário utilizar todos esses bytes, pois, se assim não fosse o processo descrito no parágrafo anterior, não funcionaria correctamente.

O último parâmetro, é relativo aos dados enviados na trama, estes dados serão alterados dinamicamente, antes da criação e comunicação do MOb, de acordo com a mensagem que se pretenda transmitir.

Configurações UART

Como já foi explicado, estas *gateways* permitirão a adaptação de dispositivos com ligação RS-232 para uma rede CAN-Bus. Portanto, será necessário que o AT90CAN64 permita, não só a comunicação por uma rede CAN-Bus, mas também a interface com o computador central e os dispositivos de comunicação e controlo inteligente, por RS-232. Para isso, é necessária a configuração de um dos canais UART do microprocessador. Esta configuração, deve ser igual à da porta série da interface, ou seja, 8 bits de dados, sem paridade, um "*stop bit*" e a velocidade que se utilizou foi de 115200 bits por segundo, para conseguirem acompanhar a velocidade da rede CAN, de 100kbits/s. Estas configurações e a comunicação de dados pela porta série serão realizadas, recorrendo também a uma biblioteca disponibilizada pela Atmel.

Implementação

A função destas *gateways*, é de receber uma mensagem pelo canal CAN e envia-lo para o canal UART, ou receber uma mensagem pelo canal UART e envia-la pelo canal CAN. Para isso, estas *gateways*, devem conseguir interpretar as tramas do protocolo que recebem que por um ou por outro canal e rescrevê-las para o protocolo do outro canal. O funcionamento geral destas *gateways* CAN – RS-232 está representado na figura 5.14.

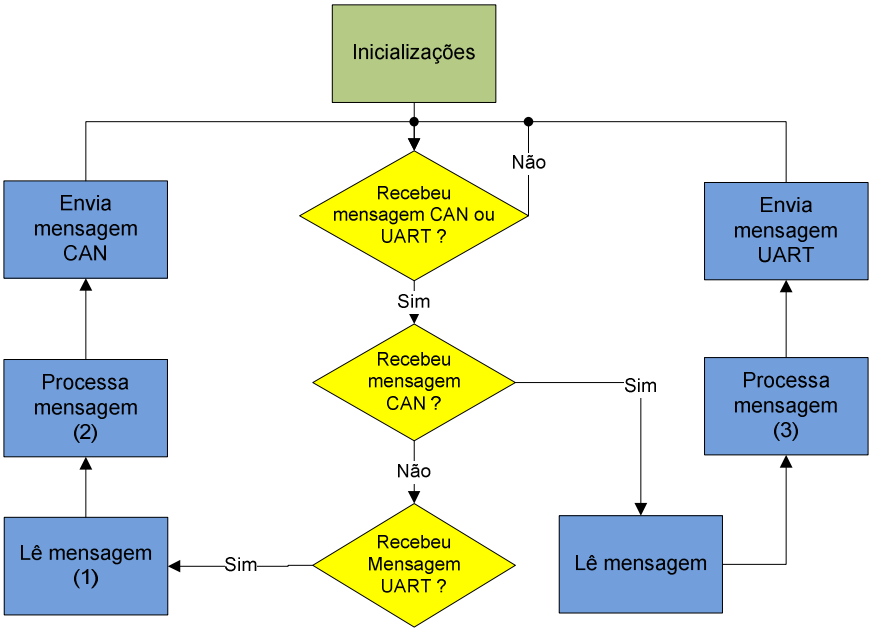


Figura 5.14 - Processo geral das *gateways* CAN - RS-232

Depois de realizar todas as inicializações/configurações o processo do microprocessador passa para um estado de espera de recepção de qualquer mensagem, seja uma mensagem UART seja uma mensagem CAN. Recebida uma mensagem, é verificado que tipo de mensagem é essa que posteriormente é lida e tratada adequadamente até ser enviada na especificação contrária à que recebeu.

Uma trama UART para ser aceite, tem de passar por um processo de reconhecimento da qualidade da mensagem, realizada no estado de leitura da mensagem, como está em cima apresentado. Esta verificação é um pouco à imagem da realizada na interface HMI. Este processo está representado na figura 5.15.

Verifica-se que o primeiro caracter recebido é um '~', e de seguida averigua -se se o CRC presente na mensagem, é o mesmo que o CRC calculado para essa mensagem. Se alguma destas duas verificações não for bem sucedida, a mensagem é ignorada, e a *gateways* volta para o estado de espera de uma mensagem CAN ou UART.

No caso das mensagens CAN, todas as mensagens recebidas são aceites, pois esta rede já faz por si só vários mecanismos de detecção de eventuais erros ocorridos na transmissão das mensagens e não há necessidade de realizar mais. Ainda nestas mensagens, para as receber, é constantemente criado o *Message Object CAN_message* recebido em cima até que se receba uma mensagem CAN de forma não mascarada. Depois de recebida uma primeira trama CAN, esta é analisada e pelo tipo de trama, percebe-se se há mais alguma trama a receber ou não, para a mensagem em causa, e se houver essa, ou essas mensagens, são recebidas de forma mascarada.

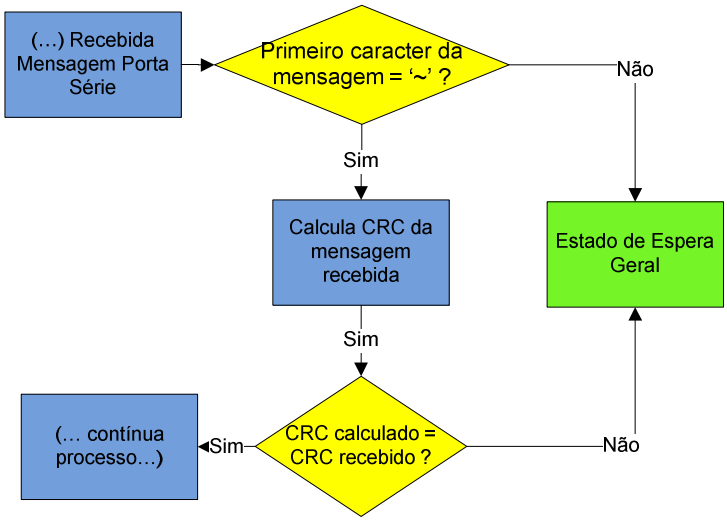


Figura 5.15 - Processo de recepção de mensagem pela porta séria na *gateway*

O processamento das tramas recebidas pela porta série é um processamento algo complexo. A trama UART é "partida" sucessivamente pelos caracteres '#', até se obterem todos os dados recebidos na mensagem, que serão posteriormente tratados para se construir a trama CAN correctamente para ser depois enviada. Todo este processo está representado na figura 5.16, repare-se que são desde logo, de modo geral, obtidos os dados relativos ao identificador do Mestre, do Escravo e da Mensagem, seguidamente é obtido o tipo de trama, sendo que, todo o restante processamento, depende desse tipo de trama adquirido. O processamento representado na figura 5.16 é relativo a uma *gateway* de um Mestre. No caso do Mestre Central, o processo é muito idêntico, havendo apenas diferenças nas tramas tipo R, que não possuem mais dados depois do parâmetro tipo de trama, e na trama tipo A, que não apresentam o estado do Actuador por ser apenas o pedido desse estado.

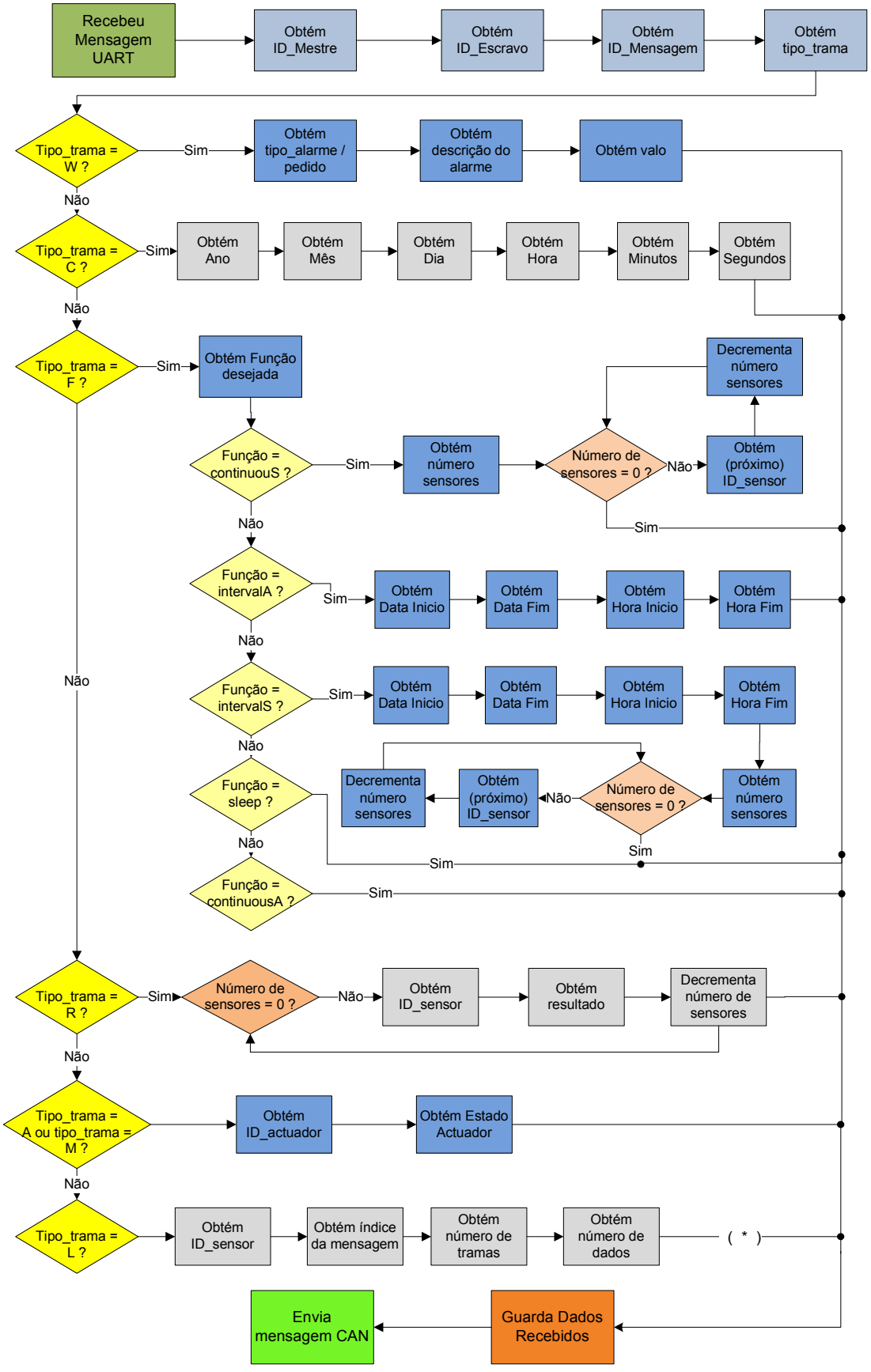


Figura 5.16 - Processamento da trama UART na *gateway*

Note-se que no esquema, é indicado que para as tramas tipo L, depois de obter todos os dados, a *gateway* passa para o estado de os guardar e enviar a mensagem CAN, com uma restrição, representada pelo símbolo "(*)". Essa restrição significa que se o índice da mensagem tipo L for 2, a *gateway* passa para um estado de espera das tramas de dados, fazendo um processamento muito diferente deste aqui indicado.

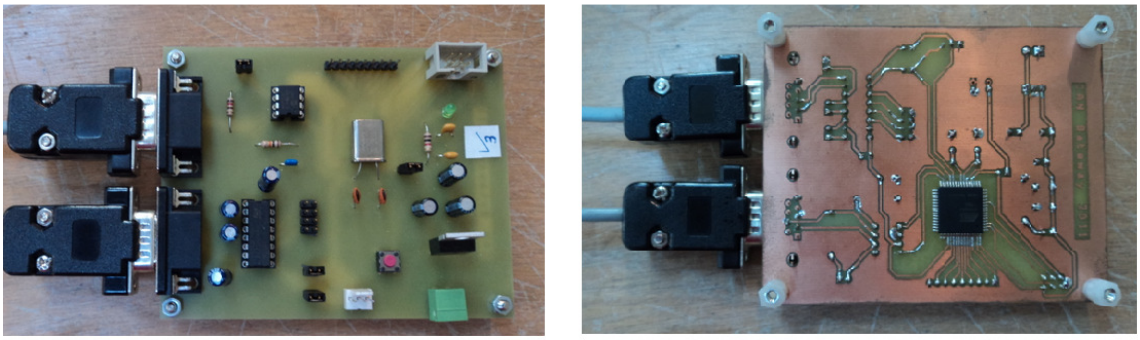
No caso do processamento das tramas CAN, os dados das mensagens recebidas, têm uma posição fixa (bytes), definida pelo protocolo CAN, esses dados são guardados e enviados por UART, seguindo um processo muito idêntico ao anterior, que trata as tramas a partir do valor do tipo de trama verificado.

Mais complexo é o algoritmo necessário para as tramas longas. A partir dos dados recebidos pela trama de índice 2, as *gateways* do lado do emissor e do lado do receptor, calculam as tramas CAN, necessárias para enviar os valores das tramas UART. Por exemplo, para tramas UART com 20 dados, são necessárias 5 tramas CAN, como já foi referido. Então, quando se recebe, por exemplo, uma trama completa, os dados são repartidos por 5 tramas enviadas sucessivamente de forma mascarada. Mas na última (ou única) mensagem, podem existir menos que 20 dados e o código implementado percebe se a mensagem que vai receber por UART e deve enviar por CAN é a última (ou única), decrementando o valor das tramas que já enviou, processo idêntico ao realizado na interface. Assim, envia apenas as tramas CAN necessárias para cobrir todos os dados existentes nessa última trama.

Circuito eléctrico das Gateways CAN – RS-232

O circuito eléctrico relativo às *gateways* CAN – RS-232, é um circuito muito simples. Apresenta todos os componentes necessários para o funcionamento do AT90CAN64 (cristal de 16MHz, condensadores, resistências, etc.), apenas há a destacar: a utilização de um conector para a programação dos microprocessadores (pinos da comunicação SPI, VCC, GND e *Reset*), que se realiza recorrendo ao programador AVR ISP; o *transceiver* e o conector RS-232, utilizados para comunicar com o computador, sendo que para o caso dos Waspmotes, não há necessidade de utilizar um *transceiver*, pois esta pode ser realizada directamente por UART, mas este é parte integrante do circuito; e por fim o *tranceiver* e o conector CAN.

Nas figuras 5.17a e 5.17b, pode ser visualizado o circuito da *gateway* em placa de circuito impresso, na sua forma definitiva. Estas placas foram desenvolvidas em conjunto com o colega de projecto.



a) b)

Figura 5.17 - PCB do circuito das *gateways* CAN - RS-232 - a) vista de cima e b) vista de baixa

5.3 - Redes de Comunicação Sem Fios

5.3.1 - Fundamentação Teórica

Como foi anteriormente mencionado, 25% do custo total e 75% do tempo de instalação de sistemas de monitorização de estruturas, deve-se à colocação de cabos para as redes de comunicações. É então muito fácil, notar a carência de uma tecnologia que permita reduzir esse custo verificado. Uma solução muito interessante para estes problemas passa pela utilização de redes sem fios. Entre outras vantagens, estas redes apresentem um baixo custo de desenvolvimento, manutenção e expansão, bem como facilidade de instalação.

Apesar da tecnologia de comunicações sem fios já estar escolhida antes da realização do projecto, nesta secção pretende-se efectuar uma pequena análise às tecnologias existentes no mercado, em particular, à tecnologia utilizada IEEE 802.15.4, justificando a sua utilização.

A tecnologia WLAN (*Wireless Local Area Network*) mais utilizada no mercado é a Wi-Fi (IEEE 802.11) que opera na banda ISM (*Industrial, Scientific and Medical*) não licenciada dos 2.4GHz (de 2400 a 2483.5 MHz), com taxas de transmissão de dados próximos dos 11 ou 54 Mbps [23].

Existem no entanto, outras soluções como as tecnologias *bluetooth* (IEEE 802.15.1) ou a tecnologia *ZigBee* (IEEE 802.15.4).

Rádio Frequência

A transmissão de dados por rádio frequência, realiza-se a partir de uma tecnologia desenvolvida durante a 2ª Guerra Mundial, a *Spread Spectrum*. Esta foi desenvolvida para dotar as comunicações entre os militares de grande segurança, de forma a impedir que estas fossem interceptadas. Esta técnica consiste em espalhar o sinal ao longo de um determinado número de diferentes frequências, sendo estas frequências pré-estabelecidas.

Esta tecnologia, apresenta certas características que a tornam ideal para comunicações sem fios, entre as quais, a possibilidade de coexistência entre sistemas rádio sem perturbação da sua actividade, bem como a possibilidade de operação na banda ISM [23].

Wi Fi – IEEE 802.11

Em 1997, foi publicada no IEEE a norma 802.11, que especifica uma forma de comunicação de dados, que utiliza o ar como meio de transmissão, não necessitando por isso de cabos eléctricos para o fazer. Esta norma, especifica as camadas mais baixas do modelo OSI, explicado anteriormente, a camada física e a camada de ligação, mais especificamente a camada MAC.

Ao nível da camada física, a norma especifica cinco técnicas de transmissão de dados possíveis: 3 métodos que utilizam a tecnologia RF, situados na banda não licenciada ISM e um método por infra-vermelhos (IR).

A técnica IR permite uma transmissão não direccional, com velocidades de 1 ou 2 Mbps, mas apresenta uma grande desvantagem, por não permitir penetração em paredes.

Uma das técnicas de RF, é a FHSS (*Frequency Hopping Spread Spectrum*), que utiliza 79 canais de frequência, situados na banda ISM. Possui uma semente, que lhe permite escolher em vários instantes, um canal aleatório de frequência a utilizar para a sua transmissão de dados. Se todos os elementos da rede tiverem a mesma semente e estiverem temporalmente sincronizados, utilizarem sempre os mesmos canais de frequência, a comunicação fluirá

naturalmente. Esta técnica está actualmente a cair em desuso, por apresentar uma baixa largura de banda.

Em 1997, apenas os métodos, até aqui enumerados, pertenciam à norma, mas em 1999 duas novas técnicas foram introduzidas: na DSSS (*DSSS Direct Sequence Spread Spectrum*), os bits são transmitidos por uma sequência de modulação de mudança de fase, que pode ser utilizada em 13 canais com uma frequência de transmissão de 5 MHz; e a OFDM (*Orthogonal Frequency Division Multiplexing*) que é uma técnica para redes de alta velocidade, pode ser utilizada nas bandas ISM de 2,4 GHz e 5 GHz. Estas duas últimas técnicas, podem ser utilizadas para velocidades muito superiores aos Mbps da norma original.

Segundo o IEEE 802.11, uma WLAN é composta por 4 componentes: Estação - componente que efectua uma ligação ao meio e pode comunicar; BSS (*Basic Service Set*), que é um conjunto de estações que comunicam entre si; ESS (*Extended Service Set*), que representa um conjunto de BSSs com pontos de acesso (AP) que lhes permitem comunicar umas com as outras; e por fim o DS que é o mecanismo que permite que os APs troquem informações mutuamente.

Na camada de MAC é utilizada uma das duas seguintes técnicas a DCF (Distributed Coordination Function) e a PCF (Point Coordination Function).

O método de operação DCF não utiliza nenhum controlador central, ao contrário do PCF. Este utiliza a técnica CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance), que executa a seguinte conduta: sempre que uma estação quer transmitir escuta o meio, se ele estiver livre a estação transmite sem interrupção, se não espera pela sua libertação para poder transmitir os seus dados. Em caso de colisão de dados estação interrompe a sua transmissão, espera um determinado tempo e tenta enviar os seus dados novamente.

Neste protocolo estão definidos três tipos de tramas diferentes: tramas de dados, tramas de controlo e tramas de gestão. Cada uma composta por um cabeçalho específico r por um conjunto de campos utilizados pela camada de ligação (MAC). Podem ainda existir cabeçalhos relativos à camada física, que estão normalmente ligados à técnica de transmissão utilizada.

Para garantir a segurança na comunicação, esta norma especifica mecanismos de autenticação, confidencialidade, controlo de acesso e integridade dos dados, para impedir a utilização dos seus serviços a utilizadores não autorizados [23].

Bluetooth – IEEE 802.15.1

A norma IEEE 802.15.1, pertence tal como a 801.15.4 que será posteriormente apresentada, à família dos *standards* 802.15, que específica no modelo OSI as camadas físicas e de ligação. As principais características de comunicações desta família são o baixo custo, o baixo consumo de energia e serem utilizadas apenas para comunicações a curta distância.

A norma 802.15.1, publicada em 2005 não é mais que uma normalização de comunicações que utilizam a tecnologia *bluetooth*. Tecnologia esta, que para além, das características atrás indicadas, apresenta: elevada robustez; baixa complexidade na sua implementação; capacidade de comunicação entre dispositivos de diferentes fabricantes, a partir da sua arquitectura de rádio e de uma camada de software que permite o desenvolvimento de aplicações, que correm em dispositivos *bluetooth* localizar outros dispositivos numa área envolvente e utilizar os serviços que estas possam oferecer; e finalmente a sua utilização na largura de banda dos 2.4 GHz não licenciada da ISM.

A norma 802.15.1 de tecnologias *bluetooth*, é utilizada essencialmente em produtos como: computadores portáteis, PDAs, telefones celulares, e outros dispositivos móveis [28].

O modelo de camadas utilizado neste *standard* não é exactamente o modelo OSI mas pode facilmente ser relacionado com este, como se verifica na figura 5.18.

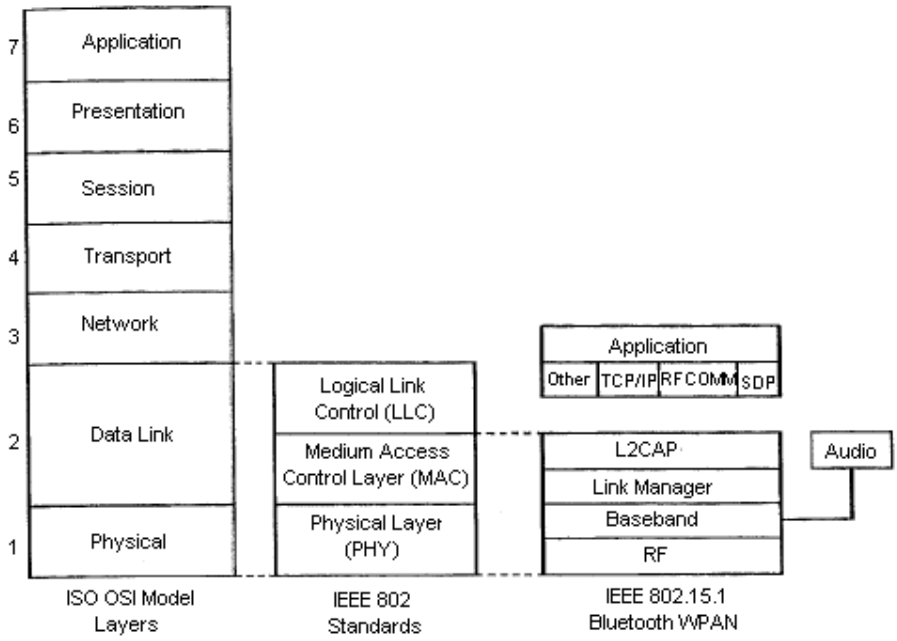


Figura 5.18 - Relação entre os modelos OSI, IEEE 802 e IEEE 802.15.1 [28]

A norma *bluetooth* está dividida em duas partes: o núcleo e os perfis. Sendo a primeira parte constituída pelos quatro níveis mais baixos do modelo (L2CAP, *Link Manager*, *Baseband*, e RF) e pelo serviço SDP (*Service Discorevy Protocol*).

A camada RF, ou camada de rádio, especifica as características de rádio, bandas de frequências de funcionamento, arranjo de canais e os níveis de potência de transmissão e recepção. A camada de *Baseband* realiza as tarefas de descoberta dos dispositivos, formatação de ligações e da comunicação síncrona ou assíncrona entre os dispositivos. O nível de *Link Manager* é responsável por transportar as mensagens necessárias em qualquer comunicação de controlo, configurações e gestão das ligações. O L2CAP (*Logical Link Controlol and Adaptation Protocol*) é semelhante à camada de ligação do modelo OSI, é responsável pela entrega dos pacotes recibos dos níveis mais elevados ao outro lado da ligação. O SDP implementa os mecanismos que permitem a um dispositivo descobrir serviços oferecidos por outro.

Os pacotes utilizados na comunicação *bluetooth* são divididos em três entidades: o código de acesso, utilizado para sincronização temporal, compensação de *offset*, inquirição e numeração; o cabeçalho, que contém a informação para reconhecimento do pacote, o número de pacotes transportados, o controlo do fluxo, endereço do dispositivo e correcção de erros do próprio cabeçalho; e por fim o corpo, que contém os dados que se pretendem enviar.

A segunda parte da norma, os perfis, são os serviços que permitem a este tipo de comunicação utilizar uma grande variedade de aplicações. Estes, especificam não só parâmetros do controlo, mas também as características e os procedimentos necessários para a interligação de dispositivos *bluetooth*.

Nesta tal, como em todas as comunicações realizadas por ondas de rádio, é fácil de perceber que os dados trocados podem ser acedidos por intrusos que possuam equipamentos

adequados. Para evitar essas situações são utilizados mecanismos de segurança específicos, tais como: encriptações dos dados e rotinas de autenticação.

ZigBee – IEEE 802.15.4

Uma outra norma da família 802.15 é a IEEE 802.15.4 desenvolvida em 2003. Esta específica a camada física e a de ligação de dados da *stack* do modelo OSI [29].

Esta norma é uma solução para comunicações sem fios de baixo custo e com muito baixo consumo de energia, especialmente indicada para aplicações como: o controlo de casas e edifícios; controlo industrial; segurança; electrónica de consumo; periféricos de computadores; monitorização médica; controlo de brinquedos ou jogos; entre outras aplicações que têm como particular carências a fiabilidade, facilidade de adição ou remoção de novos nós à rede, mas principalmente que requerem elevada duração das baterias.

Para dotar esta norma de mais funcionalidades a ZigBee Alliance em 2004, adicionou-lhe mais duas camadas superiores, a camada de rede e de aplicação como se pode verificar pela arquitectura em baixo apresentada (figura 5.19).

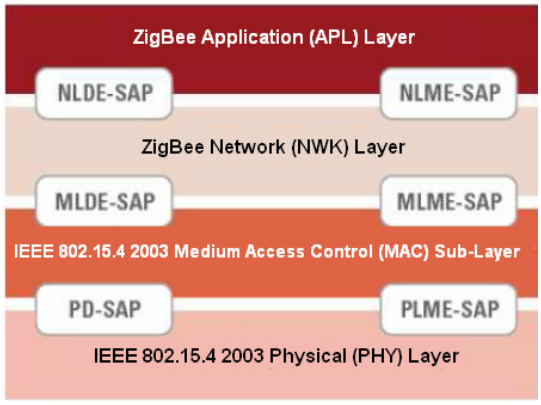


Figura 5.19 - Arquitectura da *stack da norma IEEC 802.15.4 - ZigBee* [23]

Ao nível da camada física estas comunicações funcionam nas bandas de frequência da ISM de 2,4 GHz ou nas bandas de 868 MHz na Europa e 915 MHz na América do Norte. Possuem uma largura de banda máxima de 250Kbps e utilizam normalmente a DSSS como forma de modulação. Esta camada na norma apresentada tem como principais funções: a activação e desactivação do *transciever* de rádio, que pode estar em modo de transmissão, recepção ou *sleep*; adaptação da onda ao valor energético necessário, indicação da qualidade da transmissão, avaliação da actividade do canal; e selecção do canal de frequência para a transmissão dos dados.

A sub-camada MAC suporta dois métodos possíveis para controlo do acesso ao meio: o método CSMA/CA *unslotted* e o método por *beacons*. No primeiro o nó que inicia uma transmissão de dados introduz um tempo de espera para a terminar, que permite evitar colisões caso outros dispositivos tentem aceder ou enviar mensagens ao mesmo tempo. No método por *beacons*, que são pequenas mensagens enviadas pelo coordenador periodicamente para sincronizar e identificar os nós associados a ele, após o envio do *beacon* existe um período de tempo em que os dispositivos podem enviar as suas mensagens com o meio controlado por um mecanismo CSMA/CA e um período de tempo seguinte, em que os

dispositivos podem entrar em *stand-by,* para poupar energia, enquanto esperam o próximo *beacon*.

A camada de rede é responsável por funções de gestão da rede, entre os quais a configuração dos componentes da rede, inicialização da rede, endereçamento dos dispositivos, associação, reassociação e desassociação de nós, encaminhamento das mensagens, bem como serviços de segurança da rede.

Já a camada de aplicação possui componentes que realizam a interface entre esta camada e a camada inferior (rede), objectos de aplicação que permitem realizar as aplicações pretendidas aos componentes ZigBee, por exemplo definir o tipo de elemento (coordenador, *router* ou *end device*).

Assim uma trama 802.15.4 possui um total de 11 + (4 a 20) + n bytes de dados [30], sendo os 11 bytes fixos referentes a dados de controlo e dados para verificação de erros, 4 a 20 bytes para identificação do dispositivo transmissor e do dispositivo de destino, permitem o controlo de acesso ao meio e por fim, *n* bytes referentes aos dados que se pretende enviar. Para além destes, são ainda recebidos mais 11 bytes de *acknowledge* do dispositivo de destino se a transmissão for realizada com sucesso.

Ao contrário da norma 802.15.1, esta norma foi desenvolvida para aplicações com muitos componentes e que transmitam tramas de dados muito pequenas. Com estas características facilmente se verifica que esta rede pode ser utilizada como rede de campo de um sistema de comunicações industriais.

Numa rede ZigBee existem 3 componentes: o coordenador (ZC), o *router* (ZR) e os *end devices* (ZED).

O coordenador, que é único, controla a WPAN, pois tem a responsabilidade de iniciar a configuração dessa rede, que após finalizada, este pode exercer funções de *router*.

O *router*, que é responsável pelo encaminhamento das mensagens na rede, pode também ainda associar-se ao coordenador da rede a outros *routers*, ou a *end devices*.

O *end device*, está associado a um único *router*. É apenas um nó de sensores e/ou actuadores.

As três topologias possíveis para o *standard* 802.15.4 estão representadas na figura 5.20.

Na topologia em estrela, existe apenas um nó que opera como coordenador. O paradigma de comunicação na topologia em estrela é centralizado, isto é, cada dispositivo da rede que pretenda comunicar outro dispositivo, deve enviar ao coordenador a mensagem que a entregará ao nó de destino pretendido [31]. Não é uma topologia muito aconselhável para redes de sensores *wireless,* porque o coordenar por estar constantemente a ser invocado vai rapidamente gastar a energia das suas baterias, a não ser que o ZC, seja alimentado por uma fonte fixa.

Na topologia *mesh*, tal como na anterior, existe apenas um coordenador, no entanto aqui a comunicação é feita de forma descentralizada, podendo qualquer nó comunicar directamente com outro desde que esteja ao seu alcance. Esta topologia permite mais flexibilidade, contudo, induz uma complexidade adicional para conectar todos os nós existentes na rede. Numa rede em *mesh* a eficiência energética é maior, bem como a utilização pelos nós das suas baterias é mais "justa" quando comparada com a topologia em estrela, porque a comunicação não é dependente de um nó em particular.

A topologia *cluster-tree* é um caso particular da topologia *mesh*, mas onde existe apenas uma caminha possível para cada qualquer par de nós. O processo de comunicação consiste

num *end device*, enviar a sua mensagem ao router que lhe está associado, que encaminha essa mensagem de router em router até ao *end device* de destino pretendido.

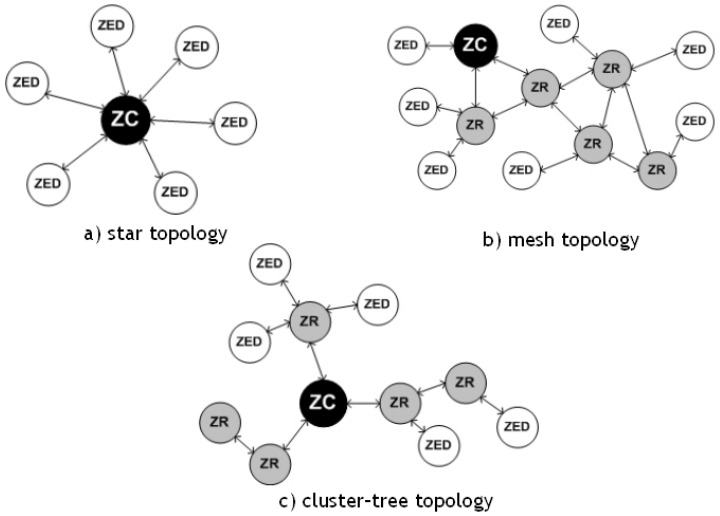


Figura 5.20 - Topologias de redes segundo IEEE 802.15.4: a)estrela, b) *mesh* e c) *cluster-tree* [31]

Comparação entre Tecnologias de Redes Sem Fios

Numa visão geral comparando estes protocolos de comunicação sem fios, pode referir-se que, uma rede de comunicações sem fios, segundo a norma IEEE 802.11/Wi-Fi, deve ser utilizada quando se pretendem transmissões de grandes quantidades de dados; a norma IEEE 802.15.1/Bluetooth para troca de dados de quantidades intermédias; e a norma IEEE 802.15.4/ZigBee para comunicações onde a quantidade de dados por pacote seja muito reduzida.

Na tabela 5.2 podem verificar-se as principais características destas redes de comunicação *wireless*.

Tabela 5.2 — Protocolos de Comunicação *Wireless* [32]

	IEEE 802.11 Wi-Fi	**IEEE 802.15.1 Bluetooth**	**IEEE 802.15.4 ZigBee**
Aplicações	Aquisição de dados para PCs	Periféricos de PCs, Sensores	Sensores Wireless, Automação
Alcance	30 a 100 m	10 a 30 m	50 a 100 m
Largura de Banda	54 a 540 Mbps	1 Mbps	250 Kbps
Frequência	2.4 GHz, 5 GHz	2.4 GHz	900 MHz, 2.4 GHz
Duração das Baterias	Horas	Dias	Anos
Segurança	Melhor	Bom	Muito Bom

5.3.2 - Implementação das Comunicações *Wireless* 802.15.4

Como neste sistema protótipo existem apenas um ou dois dispositivos de campo, associados a um único dispositivo de comunicação e controlo inteligente, os quais comunicarão entre si por redes sem fios ZigBee 802.15.4, a topologia de distribuição destes dispositivos na rede sem fios é em estrelam, pois, não faz sentido utilizar uma topologia com dispositivos a funcionar como *router*, quando nestas redes apenas existem, no máximo, 3 módulos. Assim, existirá um coordenador que será a Unidade de Comunicação e Controlo Inteligente (Mestre), e um ou dois *end-devices*, que serão as Unidades de Processo (Escravos).

Os XBEEs têm, para funcionar correctamente, de ser devidamente configurados. Existem então, parâmetros a ter em conta na configuração de uma rede sem fios, tais como:

- O endereço MAC dos módulos XBEE - este endereço de 64 bits, é único para cada módulo e não pode ser modificado, sendo atribuído durante a sua produção pelo fabricante;
- Endereço na Rede – é um valor de 16 bits, que permite identificar o módulo na numa rede a definir;
- Endereço da Rede (PAN ID) – é um endereço de 16 bits, utilizado para identificar a rede a que os módulos pertencem;
- Identificador do Nó – é um conjunto de caracteres (máximo 20), que permite atribuir um nome a cada módulo (nó) da rede;
- E por último o Canal – é relativo ao canal utilizado na banda de frequências ISM de 2.40-2.48GHz, que contém 16 canais possíveis para a comunicação.

Esta configuração pode ser realizada utilizando o API dos Waspmotes ou através de um software específico da Digi (fabricante dos componentes). No caso do sistema implementado, utilizou-se o software da Digi e configuraram-se os módulos com parâmetros escolhidos sem nenhum critério específico. Apenas de realçar que se configurou apenas uma rede e não duas, como se poderia imaginar, visto haver dois Mestres a controlar redes distintas. Esta opção, foi a definida por permitir também a possibilidade, não implementada, de comunicação também entre os Mestres, por exemplo.

Depois de configurada a rede nos XBEE é necessária agora a sua inclusão nos Waspmotes. Esta inclusão é muito simples, e é realizada tanto ao nível do hardware, como ao nível do software. A inclusão nos Waspmotes dos XBEEs por hardware, passa por conectar os dispositivos de comunicação *wireless* na placa dos Waspmotes, por sua vez, a inclusão por software, requer uma configuração dos Waspmotes. Esta configuração é realizada de forma muito simples pois o API do Waspmote já contém um conjunto de bibliotecas que permite essa configuração, sendo necessário apenas indicar ao Mote o protocolo de comunicação utilizada, os módulos que realizarão a mesma e por fim a banda de frequência utilizada nesta comunicação. Neste caso, trata-se de uma comunicação por IEEE 802.15.4, utilizando módulos XBEE normais, na banda de frequência de 2.4GHz. Introduzidos estes dados, é agora apenas necessário ligar, por software, o XBEE para que a comunicação *wireless* se possa concretizar.

Configurações UART

É ainda necessário configurar a porta UART do Waspmote, para permitir a sua ligação com as gateways CAN – RS-232. Esta configuração deve ser igual à utilizada nas gateways de modo a permitir uma troca de dados correcta. Utilizam também nesta configuração, as bibliotecas

do API do Waspmote que permitem efectuar também de forma muito simples essa configuração. Define-se então para a porta UART: largura de banda de 115200 bits/s; 8 bits para transmitir um dado; sem paridade; e um único *stop bit*.

5.4 - Implementação das Unidades de Comunicação e Controlo Inteligente

Explicada toda a matéria relativa às comunicações entre os vários elementos do sistema, serão seguidamente explicados todos os processos relativos ao funcionamento das Unidades de Comunicação e Controlo Inteligente do sistema (Mestres).

Estes algoritmos e implementações foram desenvolvidos, em trabalho conjunto com o colega de projecto.

Processo Geral

Na figura seguinte (figura 5.21), está representado num fluxograma, o funcionamento geral destes dispositivos.

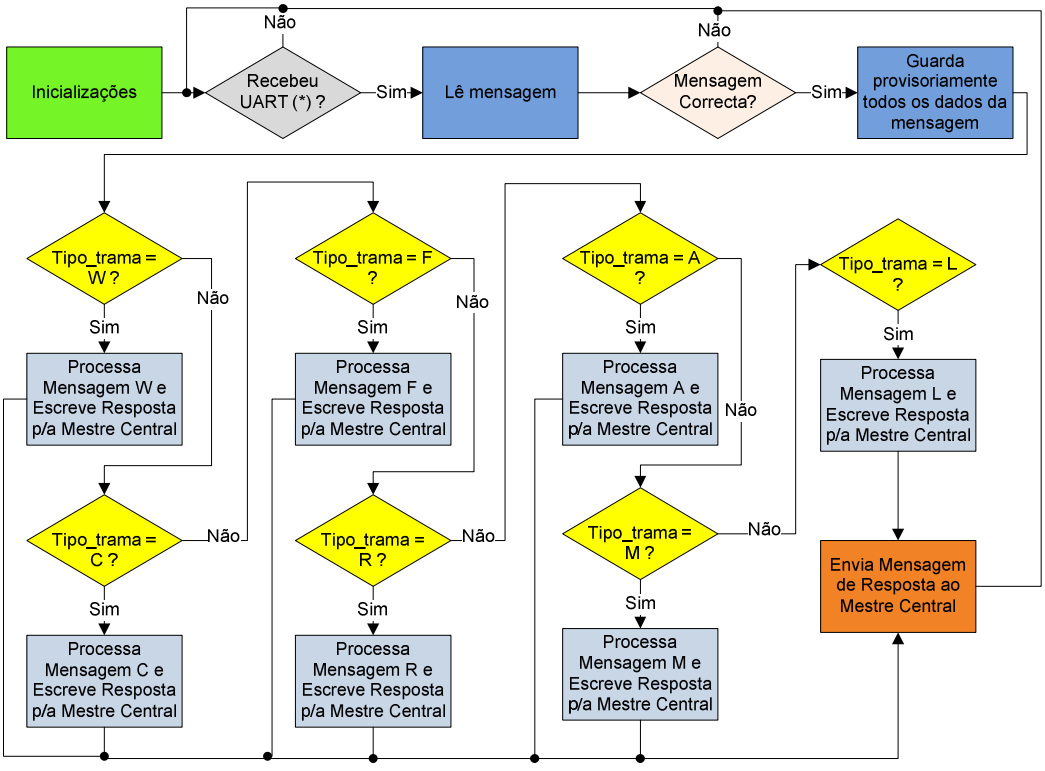


Figura 5.21 - Processo do funcionamento geral dos dispositivos de comunicação e controlo

Atente-se que, quando estes dispositivos são iniciados é realizado automaticamente um conjunto de configurações, relativas ao hardware e software destes dispositivos. Estas configurações são relativas aos módulos de comunicação sem fios, os XBEE, à porta série (UART) e também ao RTC. Este último é inicializado com valores aleatórios e começa a contar, ficando assim, à espera de ser sincronizado pela Unidade de Comando. Realizadas estas inicializações, o Mestre passa para um estado de bloqueio, à espera de uma mensagem

UART vinda da Unidade superior, ou então, aguarda que o limite de tempo de um temporizador interno seja ultrapassado, como ficará visível em seguida. Por conseguinte, quando é recebida uma mensagem pela porta UART, o dispositivo de comunicação e controlo inteligente, capta essa mensagem e verifica se é correcta, isto através de um processo em tudo idêntico ao realizado pela gateway e pela interface (figura 5.15). Se esta mensagem for assumida como correcta, é posteriormente sujeita a um processamento, específico para cada tipo de mensagem do protocolo estabelecido. Desse processamento surgirá uma mensagem de resposta que será depois enviada ao Mestre Central.

Sempre que em algum destes processamentos for enviada uma mensagem e recebida uma resposta dos Escravos, a coerência dessa mensagem é (como as mensagens recebidas por UART) testada.

Tratamento dos Alarmes

Na figura 5.22, está representado o processo relativo ao tratamento das tramas para leitura de alarmes recebidas do Mestre Central.

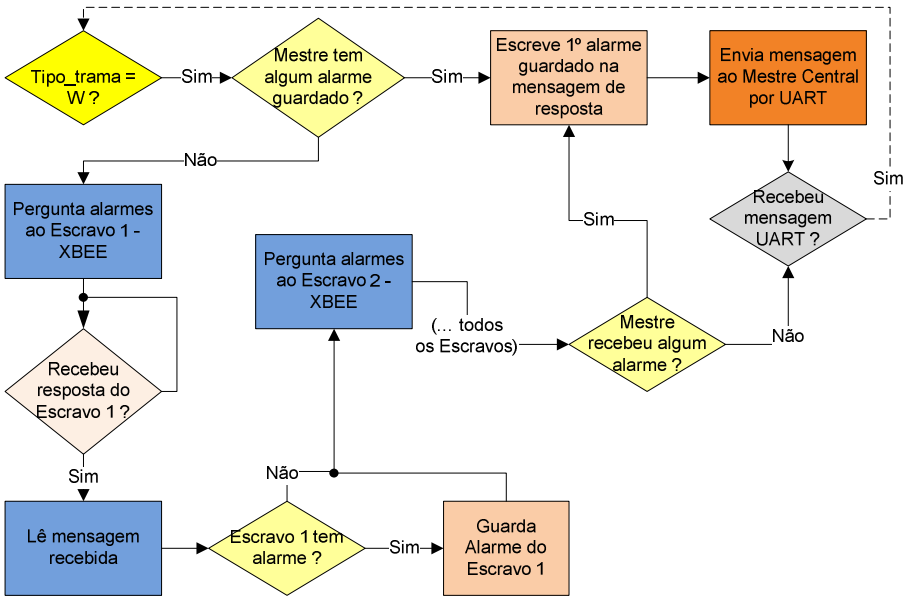


Figura 5.22 - Processamento relativo às tramas de alarmes nos diapositivos de comunicação e controlo inteligente

Pela análise da figura, pode-se constatar que, quando esta trama é recebida, o Mestre verifica na sua memória se tem já algum alarme guardado, que tenha sido obtido por leitura do mesmo nos seus Escravos. Se tiver algum alarme, o Mestre escreve a resposta a enviar ao Mestre Central incluindo o primeiro alarme da sua lista, apagando-o da sua memória em seguida. Essa mensagem é posteriormente enviada ao Mestre Central, mas note-se que não serão enviados a este, numa única resposta, todos os alarmes que possam ter ocorrido, é sim, enviado um de cada vez, por ordem cronológica de ocorrência, a cada pergunta de alarmes do Mestre Central. Se não possuir qualquer alarme na sua memória, o Mestre pergunta a um dos seus Escravos, recorrendo para tal às redes *wireless*, se este registou a ocorrência de algum alarme e em caso afirmativo guarda essa ocorrência. Contudo, independentemente de ter ou não ocorrido um alarme no primeiro Escravo, a quem a leitura de alarmes foi pedida, o

Mestre pergunta seguidamente por alarmes, a outro Escravo, isto até obter resposta de todos os seus Escravos. Neste sistema existem apenas dois escravos, mas, este algoritmo foi desenvolvido para poder ser adaptado a sistemas reais em que o Número de Escravos pode ser muito elevado. Então, depois de perguntado a todos os Escravo se ocorreram alarmes nos mesmos, o Mestre verifica na sua memória se guardou algum novo alarme e em caso afirmativo, escreve o primeiro alarme registando na resposta ao Mestre Central, sendo que, outros possíveis alarmes que tenham sido registados, serão enviados em pedidos posteriores.

Ainda, por iniciativa própria estes dispositivos enviam tramas de leitura de alarmes aos seus Escravos, da forma representada na figura 5.23, não realizando essa tarefa apenas, quando isso lhe é solicitado pelo Mestre Central. Para isso, nestes dispositivos, é configurado um temporizador que conta continuamente num período de 500ms, o que obriga o processo do dispositivo Mestre a, no final de cada contagem, sair do estado de bloqueio e ler uma possível ocorrência de alarmes nos seus Escravos. Esta tarefa difere da executada na recepção de uma trama tipo W, no facto dos alarmes, caso ocorram, serem apenas guardados na memória do Mestre Central e não enviados a este. Pois, neste sistema nenhum dispositivo tem permissão para iniciar uma comunicação com o Mestre Central.

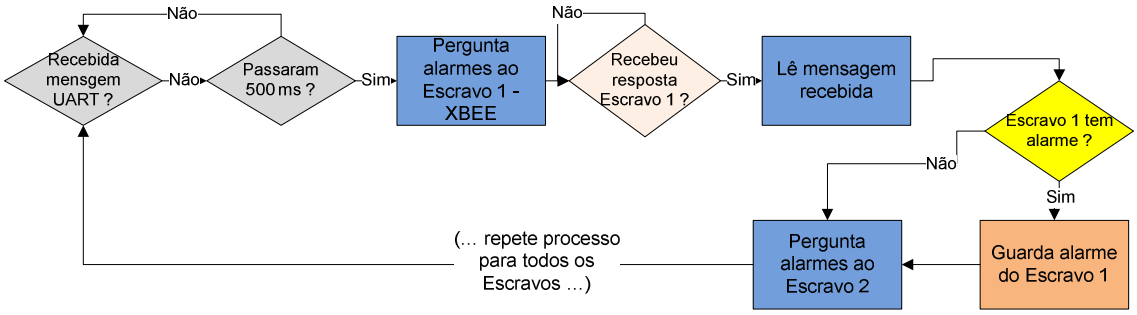


Figura 5.23 - Processo de leitura de alarmes de forma autonoma nos Escravos, pelos dispositivos de comunicação e controlo inteligente

Tratamento da Sincronização dos Relógios Internos

O tratamento das tramas de sincronização dos RTCs, recebidas pelos dispositivos de comunicação e controlo, está representado no fluxograma da figura 5.24.

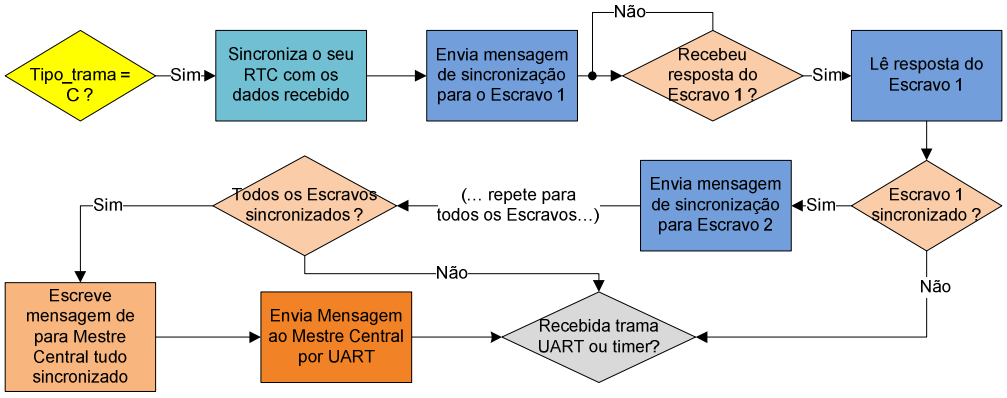


Figura 5.24 - Processamento das tramas de sincronização dos RTCs nos diapositivos de comunicação e controlo inteligente

Neste caso, a primeira tarefa que os Mestres executam é a sincronização dos seus próprios relógios internos. Feito isso, enviam pedidos de sincronização sucessivos, aos seus Escravos aguardando uma resposta de confirmação de todos eles. Quando verifica que todos os Escravos estão devidamente sincronizados, escreve e envia uma mensagem de confirmação da execução dessa tarefa à Unidade Central, passando em seguida novamente ao seu estado de bloqueio.

Tratamento a Pedidos de Leitura de Vários Valores Consecutivos de um Sensor dos Escravos Associados

O processo mais complexo relativo a estes dispositivos, é o tratamento das tramas de leitura de vários valores consecutivos de um sensor dos Escravos associados, sendo que no sistema desenvolvido, utiliza-se apenas para os acelerómetros, que são os sensores cuja informação gráfica é mais relevante. Este processo está representado figura 5.25.

O dispositivo de comunicação e controlo, é o dispositivo que impõe o número de dados e tramas a enviar, então, a primeira tarefa que realiza quando lhe é enviada uma trama tipo L, é definir e calcular os dados, o número de tramas e o número de dados por trama, ou seja o número de tramas completas e o número de dados na última (ou única) trama. Realizado isto, é então enviada a mensagem de resposta (índice 2) ao Mestre Central, onde é indicado o número de tramas e dados que serão enviados em seguida. No sistema protótipo desenvolvido, são enviados sempre 200 dados em 10 tramas diferentes, no entanto o processo implementado, está preparado para que esse valor seja dinâmico, para que possa ser utilizado num desenvolvimento futuro. Depois de enviada a trama referida, o dispositivo de comunicação e controlo inteligente fica à espera de uma resposta do Mestre Central, a qual é indicativa da recepção e aprovação desses valores, passando assim o dispositivo, por meros instantes, a funcionar como Mestre da Unidade Central. Recebida esta confirmação por parte do Mestre Central, o dispositivo inicia a aquisição dos dados do sensor pretendido no Escravo. Essa aquisição é realizada enviando uma mensagem de leitura do sensor específico, ficando depois à espera de uma resposta curta, com apenas esse valor. A cada valor recebido do Escravo, é decrementado o número de dados total a enviar até que esse valor seja nulo, ou seja, até que todos os dados sejam recebidos. Aquisição dos dados anteriores deve ser realizada com uma frequência controlada, no caso dos acelerómetros a cerca de 100 Hz. Quando todos os dados foram adquiridos, é necessário envia-los, em trama de 20 valores no máximo (como foi definido), à Unidade de Interface, Comando e Controlo que os tratara devidamente.

A escrita e envio das tramas com as medições, é realizado pelo processo ilustrado na parte inferior da figura 5.25, onde se verifica que o número de dados enviados ao Mestre Central, depende da trama a enviar ser ou não a última. Se não for a última trama, são enviados 20 dados por ordem de medição na trama. Se a trama for a última, é enviado o número de dados calculado na primeira tarefa de todo este processo, cálculo esse que é idêntico ao realizado pela Interface, representado na figura 4.11 (Sub-processos de leitura de vários valores consecutivos das medições dos sensores). Enviados todos os dados, o dispositivo fica à espera da confirmação, por parte do Mestre Central, da recepção de todos os valores, através de uma trama L com índice 3, só então é dada este processo por terminado.

Como deve ter ficado evidente, este processo implica o bloqueio quer do Mestre, quer do Escravo e até da própria rede, durante um período de tempo que se não for bem gerido e se

for usufruído pelo utilizador de forma exagerada, pode implicar atrasos significativos na entrega de mensagens prioritárias, como os alarmes por exemplo.

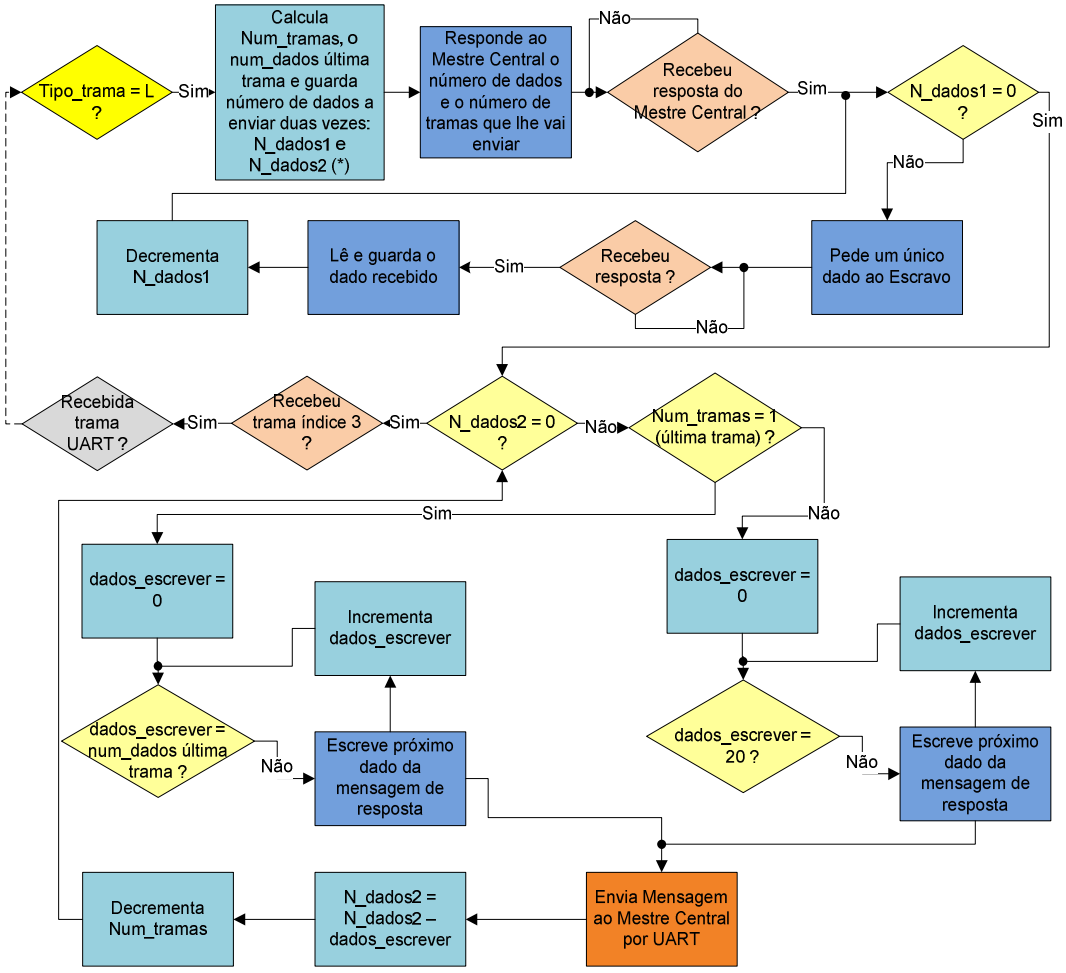


Figura 5.25 - Processamento relativo às tramas de leitura de vários valores consecutivos nos sensores dos Escravos associados aos diapositivos de comunicação e controlo inteligente

Tratamento das Tramas Direccionadas aos Escravos Associados

Por fim, falta perceber, como funcionam estes Mestres nas situações em que devem operar como ponte de comunicação entre a Unidade Central e as Unidades de Processo. Essa situação acontece quando este recebe uma trama: de definição da função de medição dos sensores dos Escravos; de leitura de valores de sensores dos Escravos associados; de leitura do estado de um actuador do Escravo; ou para mudança do estado de um actuador pertencente a um Escravo associado.

O processamento realizado pelo Mestre nestes casos (figura 5.26), passa por verificar o destinatário da mensagem pelo ID e enviar a trama ao Escravo respectivo. Seguidamente esperará uma resposta, que se verificada a sua coerência, é posteriormente encaminhada pela rede CAN-Bus ao Mestre Central.

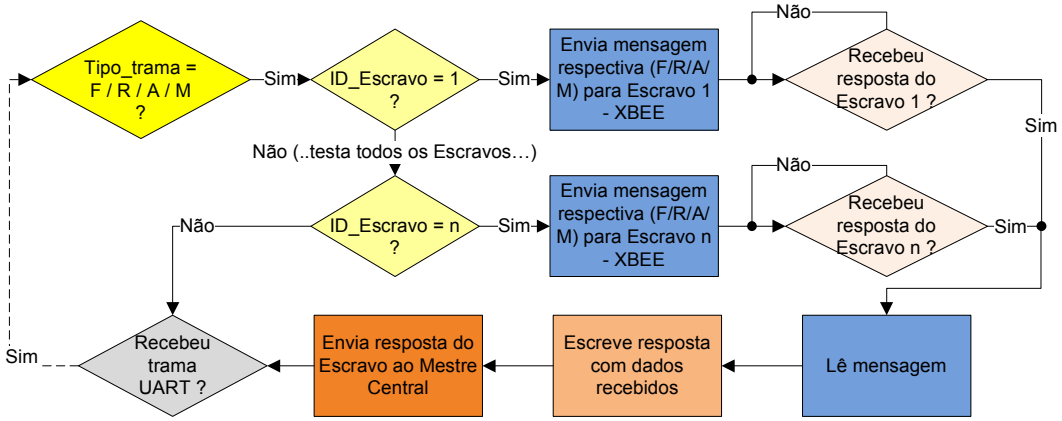


Figura 5.26 - Processo relativo ao funcionamento dos dispositivos de comunicação e controlo inteligente como ponte de comunicação entre a Unidade superior e as Unidades inferiores

Capítulo 6

Unidade de Processo

As Unidades de Processo, que podem também ser designadas de Unidade de Campo ou Unidade de Sensores e Actuadores *Wireless*, são as unidades de mais baixo nível do sistema protótipo de supervisão e controlo implementado. Estas, têm a função de recolher todos os dados do processo do sistema, que neste caso é uma estrutura física, mais especificamente uma ponte, e envia-los às Unidades Superiores (Unidade de Interface, Comando e Controlo e Unidade de Comunicação e Controlo Inteligente). Os dados aqui recolhidos, têm origem nos sensores de cada um destes dispositivos. Todavia, são tratados de maneira diferente conforme as intenções das Unidades superiores, sendo que a forma mais natural serão pedidos de leitura directos das medições dos sensores destes dispositivos. Pedidos aos quais, estas Unidade têm obrigação de responder de acordo com as intenções do operador, que pode querer ler todos ou apenas alguns sensores, e em grande ou reduzida quantidade.

Pode ainda, ser-lhes pedido que tratam os dados recolhidos dos sensores sob a forma de alarmes, isto é, com algum processamento inteligente de que foram dotados, verificar se os valores medidos nos seus sensores correspondem a alarmes. Perante tais pedidos, estas Unidades deverão ainda, verificar a sua bateria e activar esses actuadores de informação caso verifiquem que esta está com pouca carga, de modo a avisar atempadamente os responsáveis pela obra, para que procedam à sua substituição ou ao seu carregamento.

Uma outra funcionalidade destas Unidades de Processo, é a utilização de actuadores físicos, no protótipo implementado. Estes actuadores dizem respeito apenas a *leds* que representam um semáforo e permitem a demonstração do funcionamento dos actuadores nestes dispositivos.

Ainda de referir, e aliás como já deve ter ficado evidente, que estas Unidades representam na arquitectura dos sistemas SCADA, as unidades RTUs, por desempenharem todas as suas funções explicadas na secção 2.2.2.

Estas Unidades comunicam directamente, apenas com as Unidades de Comunicação e Controlo Inteligente numa rede wireless ZigBee - 802.15.4, que foi aqui configurada da mesma forma que para os dispositivos superiores, para permitir essa comunicação.

Como é evidente, não é possível ligar directamente os sensores a estes dispositivos, sendo necessário intermediariamente um circuito de aquisição e condicionamento dos sinais destes sensores, para então depois conecta-los aos dispositivos de processo. Assim, neste capítulo, será realizada uma primeira análise teórica relativa à medição de grandezas estruturais,

seguida de uma apresentação dos circuitos de aquisição implementados e na parte final a explicação dos processos de software utilizados no dispositivo de processo.

Esta Unidade, foi desenvolvida em trabalho com o colega de projecto. Já os circuitos de aquisição e condicionamento de sinal, foram desenvolvidos na sua totalidade por ele.

6.1 - Medição de Grandezas Estruturais

Na concepção de sistema de supervisão e controlo estruturais, importa perceber, quais as grandezas estruturais que importa medir e a melhor solução para o fazer, ou seja, conhecer os sistemas mais eficientes para a sua aquisição, processamento e tratamento de dados necessários, com vista a extrair a informação essencial ao estudo das condições da estrutura motorizada e ainda, compreender as formas de actuação em tempo-real e não tempo-real nessas estruturas, de forma a melhorar o seu desempenho estrutural.

Na observação de estruturas podem distinguir-se grandezas caracterizadoras tanto para o seu comportamento global como para o seu comportamento local. De entre as grandezas do comportamento global podem destacar-se deslocamentos, flechas, rotações, reacções de apoio e aberturas de juntas de dilatação. Como exemplos de grandezas de comportamento local das estruturas podem referir-se a medição de tensões (cargas), extensões e abertura de fendas. Para além destas grandezas, podem ainda ser medidas nas estruturas outras grandezas relativas à durabilidade da estrutura e às suas condições ambientais, como por exemplo a corrosão das armaduras ou medição da temperatura ambiente. O estudo da temperatura assume particular importância, uma vez que a acção térmica introduz variações de estado de deformação, tensão nos elementos estruturais e afecta também as características do próprio sistema de medição. Devem por isto, ser instalados termómetros em número suficiente, sobretudo nas secções instrumentadas [33].

Para este projecto serão medidas extensões, acelerações e temperaturas. Para isso serão utilizados: 6 extensómetros, todos conectados ao dispositivo de processo 1 (Escravo 1); 4 acelerómetros, repartidos de igual forma pelos Escravos 2 e 10; e ainda dois sensores de temperatura, também repartidos pela Escravo 2 e 10.

6.1.1 - Medição de Extensões

Os sensores destinados à medição pontual de extensões são designados de extensómetros. Existem vários tipos de extensómetros: extensómetros de resistência eléctrica, extensómetros de cordas vibrantes, extensómetros de fibra óptica e transdutores de deslocamento. E quanto ao seu princípio de funcionamento podem distinguir-se os extensómetros de fibra óptica e os eléctricos (resistência eléctrica e cordas ressonantes).

A medição de extensões numa estrutura, é na realidade, a avaliação da extensão média num segmento, cujo comprimento poderá ser maior ou menor, dependendo da homogeneidade do material a instrumentar.

Em estruturas metálicas, de madeira, de alvenaria ou em estruturas existentes, utilizam-se extensómetros de aplicação à superfície. Por seu lado, em estruturas de betão é comum aplicar extensómetros de embeber antes da realização da betonagem [13].

Na aplicação de sensores de fibra óptica na monitorização de estruturas de Engenharia Civil tem-se verificado um grande desenvolvimento nos últimos anos. As suas principais vantagens relativamente aos extensómetros eléctricos, residem na imunidade aos campos

electromagnéticos, na reduzida perda de sinal para grandes distâncias e nas suas reduzidas dimensões [34].

No entanto os extensómetros de resistência eléctrica são mais utilizados em monitorização de estrutura, especialmente por se apresentarem como uma das soluções mais económicas. Serão seguidamente detalhados por serem a solução utilizada no projecto desenvolvido.

Extensómetros de Resistência Eléctrica

Este tipo de extensómetros foi desenvolvido e fundamentado na teoria da ponte de *Wheatstone* [35]. O esquema eléctrico da ponte de *Wheatstone* está a seguir representado (figura 6.1).

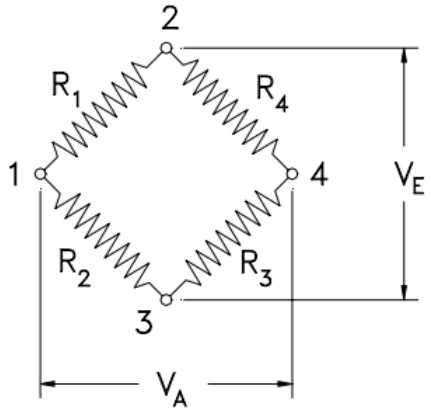


Figura 6.1 - Ponte de *Wheatstone*

Se o circuito da ponte for alimentado, entre os nós 2 e 3, por uma tensão constante V_E (tensão de entrada) aparecerá entre os nós 1 e 4 uma diferença de potencial V_A (tensão de saída), dependente do quociente das Resistências R1/R2 e R4/R3, originando a seguinte equação:

$$\frac{VA}{VE} = \frac{R1}{R1+R2} - \frac{R4}{R3+R4} = \frac{R1 \cdot R3 - R2 \cdot R4}{(R1+R2)(R3+R4)} \tag{6.1}$$

Se a ponte for equilibrada, significa que o valor das resistências é igual e o quociente V_A/V_E é nulo. Partindo deste princípio, é compreensível que qualquer variação numa ou mais resistências da ponte, provocará uma variação na diferença de potencial V_A. Ao assumir que a variação (ΔRi) é muito inferior à própria resistência (Ri), o que geralmente é sempre válido, obtém-se a seguinte relação:

$$\frac{VA}{VE} = \frac{1}{4} \cdot \left(\frac{\Delta R1}{R1} - \frac{\Delta R2}{R2} + \frac{\Delta R3}{R3} - \frac{\Delta R4}{R4} \right), \tag{6.2}$$

Ou simplesmente:

$$\frac{VA}{VE} = \frac{1}{4} \cdot (\varepsilon 1 - \varepsilon 2 + \varepsilon 3 - \varepsilon 4), \tag{6.3}$$

Os extensómetros podem ser utilizados nas pontes de *Wheatstone* de três maneiras distintas: ponte completa, meia ponte e um quarto de ponte. Estas, relacionam-se com o número de extensómetros, que podem ser vistos como resistências variáveis, utilizados, 4, 2 ou 1, respectivamente. De facto, a ponte de *Wheatstone* é sempre completa, ou seja é sempre constituída por 4 resistências, sendo parcial o totalmente formada por extensómetros.

A relação de proporcionalidade entre a variação da resistência do extensómetro e a extensão é dada pelo factor de ganho do extensómetro - G.

$$\frac{dR}{R} = G \cdot \frac{dL}{L}, \tag{6.4}$$

Sendo R a resistência de um condutor eléctrico de tamanho L. A expressão anterior pode ainda tomar o seguinte aspecto:

$$x = G \cdot \varepsilon, \tag{6.5}$$

Onde a variação da resistência está representada por x e a extensão por ε, que tem como unidade quantitativa o *Strain*, (possível tradução – deformação). Fica então visível por esta expressão, que é possível obter o valor da extensão, a partir da resistência medida no extensómetro.

Aplicação

Foi referido que os extensómetros podem ser aplicados à superfície ou embebidos na estrutura. O funcionamento destes aparelhos é em tudo semelhante. Apenas de realçar que os extensómetros de embeber apresentam algumas vantagens na medição de extensões em estruturas, uma vez que possuem protecção contra humidade e corrosão, são facilmente instalados em obra e medem extensões em zonas mais representativas do betão. Deste último facto resulta o seu maior cumprimento, uma vez que é necessário que a extensão obtida seja a média das extensões lidas do material heterogéneo e não extensões localizadas, devido a descontinuidades no interior da massa do betão [10].

A figura 6.2 representa um esquema geral dos extensómetros de colar. Estes são constituídos por uma malha de filamentos de aproximadamente 0,025mm de espessura.

A variação da resistência, de um extensómetro ideal, deveria ocorrer exclusivamente devida à deformação sofrida pelo elemento em que o sensor estivesse aplicado.

Para assegurar a qualidade dos resultados obtidos, deve ter-se em atenção algumas limitações e procedimentos. Um deles é a extensão mecânica aplicada ao condutor constituinte do extensómetro, que nunca deve exceder o seu limite de elasticidade, que está normalmente compreendido entre os 3mε e os 4 mε. Um outro factor, está relacionado com o próprio sensor e com adesivo utilizado para a colagem nas superfícies, ambos devem ser estáveis com o envelhecimento e com a temperatura. Por fim, o extensómetro deve ser electricamente isolado do material ao qual está a ser aplicado e protegido das condições ambientais, como representado em cima na figura 6.2b.

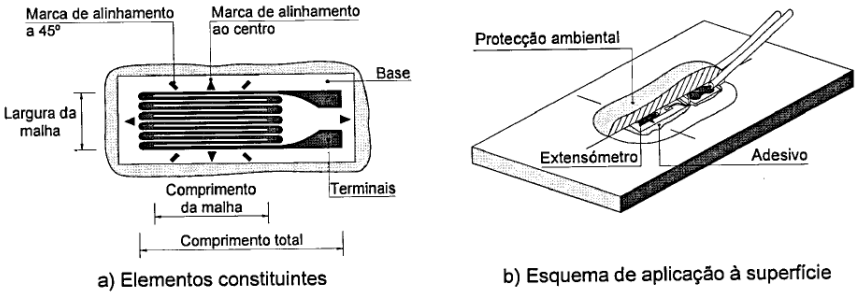


Figura 6.2 - Extensómetro de Resistência Eléctrica [34]

6.1.2 - Medição de Acelerações

A par dos extensómetros, os acelerómetros são os sensores mais utilizados na monitorização estrutural [9]. Permitem medir as vibrações/acelerações das estruturas.

Existem vários tipos de acelerómetros que diferem no seu funcionamento interno, tais como os piezoeléctricos e os capacitivos. Nesta secção, serão abordados os dois tipos destes sensores, dando particular ênfase a um tipo de acelerómetros capacitivos, os *Microelectromechanical Systems* (MEMS).

Acelerómetros Piezoeléctricos

Os acelerómetros piezoeléctricos são compostos por elementos de cristais piezoeléctricos associados a uma massa e ligados na base de uma caixa de protecção. Quando a base é sujeita a movimentos, a massa *m* exerce uma força de inércia sobre o elemento do cristal piezoeléctrico, que produz proporcionalmente a essa força, uma alteração eléctrica no cristal. Dentro de uma gama de frequências do acelerómetro, a força obedece à segunda Lei de Newton:

$$F = m \cdot a, \tag{6.6}$$

Este tipo de sensores oferece elevados campos de medidas e gamas de frequência a um custo razoável. Contudo este custo é fortemente influenciado pelos materiais piezoeléctricos utilizados. As soluções de materiais mais utilizados nestes tipos de cristais passam por cerâmicos, polímeros ou compósitos de polímeros e cerâmica, capazes de combinar as vantagens de ambos a preços moderados. Na figura 6.3 pode observar-se o esquema do princípio de funcionamento interno de um acelerómetro piezoeléctrico.

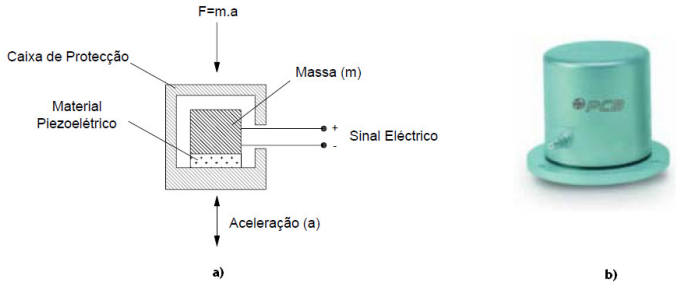


Figura 6.3 - Acelerómetros piezoeléctricos: a) príncipio de funcionamento e b) modelo PCB 393C [9]

Acelerómetros Capacitivos

Os acelerómetros capacitivos, medem acelerações, através da leitura de alterações ocorridas em condensadores eléctricos. O elemento sensor, consiste em duas placas condensadoras paralelas, actuando em modo diferencial, que opera num circuito em ponte. Por intermédio destas placas de condensadores fixas, na ocorrência de acelerações, é criada uma diferença de potencial entre elas. Alterando-se assim a tensão de pico gerada, que constituí o sinal de saída a ser processado. O seu modelo de funcionamento pode ser observado na figura 6.4a.

Estes acelerómetros funcionam normalmente a baixas gamas de frequências, no entanto o seu preço baixo, torna-os bastante atractivos e muito utilizados em redes de sensores.

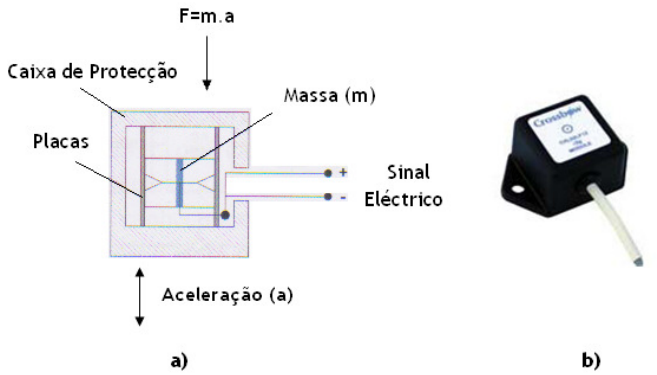


Figura 6.4 - Acelerómetro capacitivo: a) esquema de funcionamento [10] b) modelo *Crossbow* [9]

Acelerómetros tipo MEMS

A designação MEMS (*Microelectromechanical Systems*) representa hoje, todo o tipo de dispositivos eléctricos de tamanho muito pequeno. Assumem-se assim, como a junção, a electrónica à escala micro (10^{-6}) com a tecnologia de maquinaria igualmente à escala micro. É hoje uma tecnologia muito utilizada em diversas áreas, como na electrónica automóvel, equipamento médico, equipamentos electrónicos portáteis (telemóveis ou PDAs), aparelhos informáticos, entre outros. Este sucesso, deve-se a inúmeras vantagens, entre as quais o seu tamanho muito reduzido, velocidade de funcionamento maior e preços mais baixos.

Os componentes constituintes destes sistemas podem apresentar dimensões entre os 0,001mm e os 0,1mm, enquanto os sensores por si só, variam entre 0,02mm e 0,1mm. Estes sensores, apresentam num único integrado uma unidade de processamento central de dados (microprocessador) e micro sensores capazes de interagir com o exterior.

Estes MEMS apresentam num único chip, os sensores e todo o condicionamento de sinal necessário para a medição das grandezas em questão.

Este tipo de sensores, utiliza como material base o silicone, e o seu meio de produção baseia-se na deposição de sucessivas camadas, seguido de processos de fotoligrafia e de remoção selectiva de partes do condutor de silicone, ou seja, o processo é em todo semelhante ao de produção de circuitos integrados, o que à partida fornece elevados índices de funcionalidade e fiabilidade a preços relativamente baixos [36]. De facto, a expansão da utilização de micro electrónica à base de silicone, revela muitas vantagens a nível futuro, por

ser um material de fácil obtenção, elevada qualidade e baixo custo. Particularmente para a utilização em monitorização de estruturas, apresenta outras vantagens, por apresentar um comportamento perfeitamente elástico, quando submetido a esforços e a uma alta resistência à fadiga.

A unidade SI de quantificação de acelerações é o metro por segundo ao quadrado (m/s^2), no entanto é vulgarmente utilizada a unidade *g*, definida exactamente como 9,806 65 m/s^2, que é aproximadamente igual, à aceleração devida à gravidade na superfície da Terra.

Os acelerómetros do tipo MEMS da *Analog Devices*, fazem parte da gama de acelerómetros de baixo campo de medição, ou seja, para valores de aceleração de aproximadamente ±2g. Estes, apresentam uma precisão bastante alta, baixa tensão de excitação, ajuste de frequência, para diferentes aplicações, e dois ou três eixos de medição possível.

Na tabela 6.1 podem ver-se especificados dois tipos de acelerómetros, tipo MEMS da *Analog Devices*.

Tabela 6.1 — Especificação técnica de acelerómetros tipo MEMS da *Analog Devices* [10]

Modelo	Nº de Eixos de Medida	Gama de Medição (g)	Tensão de Excitação [V]	Frequência de Aquisição [kHz]	Gama de Temp. [ºC]	Dimensões [mm]
ADLX203	2	-1,7 a 1,7	3,0 – 6,0	2,5	-40 a 125	5x5x2
ADLX330	3	-3 a 3	2,0 – 3,6	1,6 (0,55 eixo z)	-25 a 70	4x4x1,45

Um esquema representativo do sensor tipo MEMS ADLX203 pode ser visualizado na Figura 6.5a.

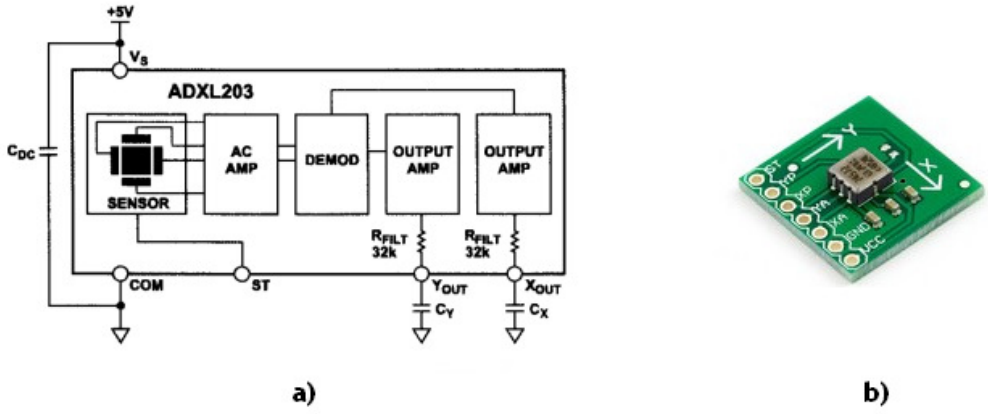


Figura 6.5 - Acelerómetro tipo MEMS ADLX203: a) esquema de ligação do sensor e b) fotografia

6.1.3 - Medição de Temperatura

Existem várias tecnologias que permitem a medição de temperatura a um sistema: termistores, termopares, RTDs (*Resistance Temperature Detector*) e sensores de temperatura em circuitos integrados (IC). Cada um com as suas vantagens e desvantagens para diferentes situações [37].

Termistores são semicondutores de resistência termicamente sensíveis, isto é, a resistência destes sensores vária com a variação da temperatura a que está sujeito. Podem ser de dois tipos: NTC (*Negative Temperature Coefficient*) ou PTC (*Positive Temperature Coefficient*), sendo os primeiros mais comuns. A diferença entre as duas variantes é que no PTC o valor da resistência medido é directamente proporcional ao valor da temperatura, ou seja o valor da resistência aumenta com o aumento da temperatura. Nos termistores NTC o processo é inverso, o valor da resistência diminui com o aumento da temperatura. Utilizam-se termistores para medir temperaturas até 150ºC, são de custo moderado e necessitam de algum condicionamento de sinal, para serem lidos correctamente.

Os termopares são a par dos RTDs, os sensores de temperatura mais utilizados. São bastantes baratos e podem medir temperaturas até 1250ºC. A sua maior limitação é a obtenção de precisão das medições, pois apresenta um valor de saída em tensão muito baixo, cerca de 40uV por ºC, necessitando de um circuito de condicionamento de sinal, algo complexo. O princípio de funcionamento destes sensores, passa pela junção de dois materiais metálicos, que geram uma tensão eléctrica em função da temperatura. Embora se admita que com qualquer combinação de 2 metais diferentes, se pode construir um termopar, utilizam-se apenas algumas combinações normalizadas, que possuem tensões de saída previsíveis. Os tipos de termopares (combinações de metais) existentes são designados utilizando letras:

- K – combinação entre Cromel e Alumel;
- E - combinação entre Cromel e Constantan;
- J - combinação entre Constantan e Ferro;
- N – combinação entre Nicrosil e Nisil;
- B, R e S – combinação entre Platina e Ródio-Platina;
- T – combinação entre Cobre e Constantan.

O termopar mais utilizado é o K, por ser possível de obter a um custo muito baixo e por permitir a medição de temperaturas entre -200 a 1200ºC, com uma sensibilidade de aproximadamente 41uV/ºC.

Os RTDs, são sensores cujo princípio de funcionamento se baseia no aumento da resistência de um metal com a temperatura. São utilizados vários metais neste tipo de sensores tais como: o cobre, ou níquel, mas o mais a utilizado é a platina, passando neste caso os sensores a ser designados por PRT (*Platinum Resistance Thermometer*). Estes sensores permitem medir temperaturas até 750ºC, são os mais precisos e têm grande durabilidade, mas são, por outro lado, caros e necessitam de um circuito de condicionamento de sinal com alguma complexidade, que normalmente incluí uma fonte de corrente precisa e ADC (*Analog-to-Digital Converter*) de alta resolução.

Os sensores de temperatura de circuitos integrados, são baseados na sensibilidade à temperatura do silício, que produzem saídas digitais e/ou analógicas. Têm o limite de medição mais baixo, no máximo de 150ºC, utilizados por exemplo para medição da temperatura ambiente. São os mais baratos e possuem já, normalmente, todo circuito de condicionamento de sinal, incluindo os conversores de sinal analógico, tornando assim a sua aplicação muito simples.

6.2 - Aquisição e Condicionamento de Sinal dos Sensores

Para aquisição dos sinais dos sensores será utilizado o ADC existente no microprocessador ATmega1281 incorporado no Waspmote. Este ADC, utiliza a técnica de aproximações sucessivas para transformar um sinal analógico num sinal digital. Neste microprocessador, a entrada é multiplexada em sete canais, permitindo assim que sejam conectados sete sinais diferentes a estes conversores. Sendo que este ADC, tem uma resolução de 10 bits e a tensão de entrada neste conversor deve estar compreendida entre os 0 e os 3.3V, obtem-se então para a tensão 0V o valor digital de 0 e para 3.3V o valor digital 1023. Este conversor, permite uma frequência máxima de conversão por cada canal de 275 kHz, que como ficará visível é suficiente para a aquisição dos sensores pretendidos.

O Waspmote possui vários pinos para a alimentação de circuitos externos, como os circuitos de aquisição e condicionamento de sinal, e para estes, permite tensões de 5V ou 3.3V, no entanto, possui uma enorme limitação neste campo, pois a corrente de alimentação máxima dos circuitos externos que este pode fornecer é de 200mA. Esta limitação, implica que todos os sensores, actuadores e circuitos relacionados com estes, tenham um consumo muito reduzido, senão será impossível com estes microprocessadores alimentar esses circuitos. Perante isto, uma hipótese para esse sistema passa por obter alimentação de outra fonte, como baterias externas. Deste modo, no sistema implementado para os sensores, circuitos de aquisição e actuadores utilizaram-se pequenas baterias recarregáveis de 9V.

6.2.1 - Extensómetros

Como já foi dito anteriormente, não foram adquiridos novos sensores, e os extensómetros utilizados foram extensómetros de resistência apresentado na secção 6.1.1.

Para a aquisição dos valores relativos aos extensómetros, utilizou-se o circuito de aquisição representado na figura seguinte.

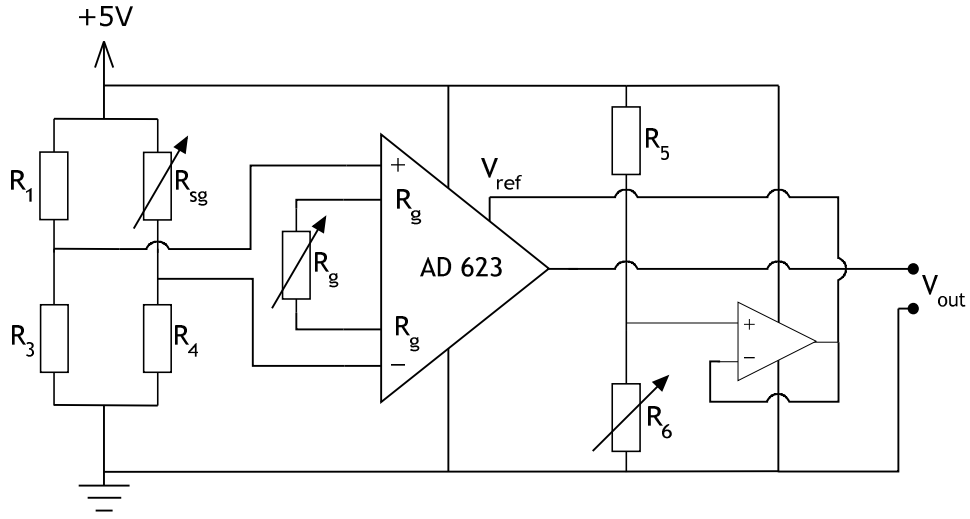


Figura 6.6 - Circuito de aquisição e condicionamento de sinal dos extensómetros

À esquerda aparece então, a ponte de *Wheatstone* com três resistências fixas iguais e uma resistência variável que diz respeito ao extensómetro. As resistências fixas de precisão utilizadas são de 350 Ω, e o extensómetro de resistência utilizado é o CEA-06-250UW 350 da

VISHAY, que apresenta um ganho de cerca de 2.1. Esta ponte de *Wheatstone* produz uma tensão de saída resultante muito pequena, pelo que se utiliza um amplificador de instrumentação, o AD 623, para amplificar esse sinal. Amplificador este que irá impor um ganho de 1000 à tensão resultante da ponte de *Wheatstone* utilizada. Por sua vez, o amplificador operacional, o OPA350, presente no circuito, permite ajustar a tensão de referencia do amplificador de intrumentação para 3.3/2 = 1.65. Sem este ajuste de referência, poder-se-ia da tensão de saída da ponte e implicativamente, da tensão de saída do amplificar, obter valores negativos, os quais não poderiam ser tratado no ADC do Waspmote, pois este apenas permite tensões de entrada entre 0 e 3.3V. Assim, com este ajuste quando a ponte *Wheatstone* estiver equilibrada, ou seja, quando não há uma extensão na estrutura, o valor obtido à entrada do ADC será de 1.65V, em vez dos 0V.

Este circuito de condicionamento de sinal foi replicado 6 vezes para os 6 extensómetros e foram implementados numa placa de circuito impresso, representada na figura 6.7.

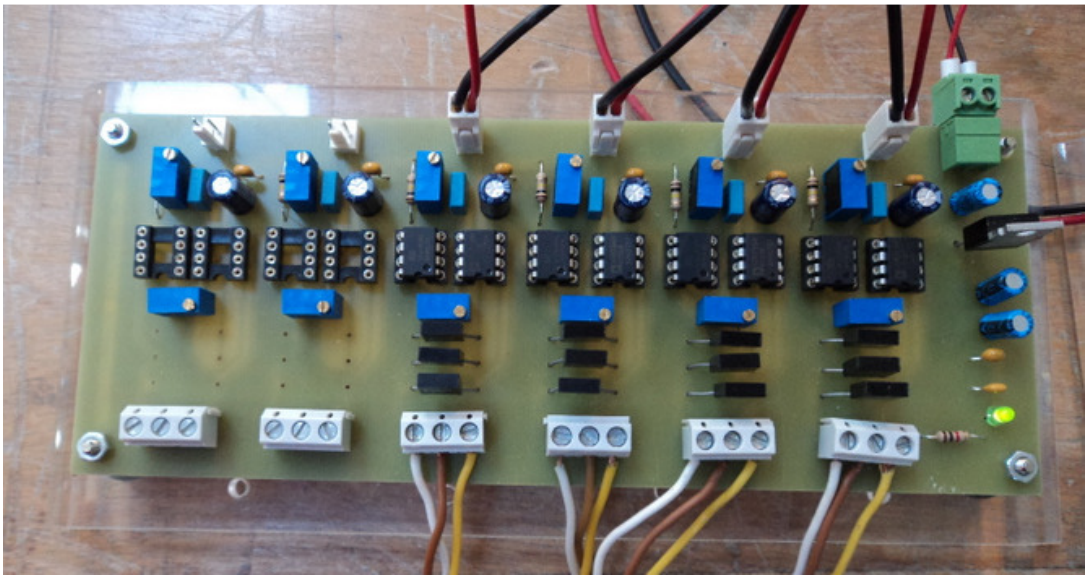


Figura 6.7 - PCB dos circuitos de aquisição e condicionamento de sinal dos extensómetros

Os valores de tensão da saída deste circuito são seguidamente convertidos em valores digitais, entre 0 e 1023, pelo ADC. Para se obter esses valores nas unidades correctas, em *Strain* (deformações), é necessário realizar um conjunto de cálculos.

Pegando então na expressão da ponte 6.1 de *Wheatstone*, e considerando que no caso, três das resistências têm valor igual (R2=R3=R4=R) e que a resistência do extensómetro tem valor variável, R·(1+x), a expressão da tensão de saída (Vo=VA) em função da tensão de entrada (Vi=VE), passa a ser:

$$\frac{Vo}{Vi} = \frac{R(1+x)}{R1(1+x)+R} - \frac{R}{R+R} = \frac{R(1+x)\cdot R - R\cdot R}{(R(1+x)+R)(R+R)} = \frac{x}{4+2x} \qquad (6.7)$$

Quando x é muito pequeno, o que normalmente se verifica para estes extensómetros resistivos, a expressão (6.7) pode ser simplificada para:

$$\frac{\mathrm{Vo}}{\mathrm{Vi}} \cong \frac{\mathrm{x}}{4} \tag{6.8}$$

Manipulando agora a equação obtida em (6.8), com a expressão (6.5), do valor da deformação em função do ganho do extensómtero, obtém-se:

$$\varepsilon \cong \frac{4}{\mathrm{G}} \cdot \frac{\mathrm{Vo}}{\mathrm{Vi}} \tag{6.9}$$

É agora, necessário ajustar este valor da tensão de saída da ponte de *Wheatstone*, para o valor de tensão de saída do circuito de condicionamento de sinal, aplicando-lhe o offset do amplificador operacional (1.65V) e o ganho do amplificador de instrumentação (1000).

$$\mathrm{Vo}' = (\mathrm{Vo} \cdot 1000 + 1.65) \tag{6.10}$$

$$Vo = \frac{\mathrm{Vo}' - 1.65}{1000} \tag{6.11}$$

Então, Vo' será a tensão de entrada do ADC, que após a conversão, originará um valor digital entre 0 e 1023. Assim, depois de convertido o resultado da medição do extensómetro no formato digital será:

$$ADC = \frac{\mathrm{Vo}' \mathrm{x}\ 1023}{V_{maxADC}} = \frac{\mathrm{Vo}' \mathrm{x}\ 1023}{3.3} \tag{6.12}$$

Portanto, a equação que permite calcular o valor da extensão na sua unidade respectiva (*Strain* - deformação), obtem-se trabalhando as expressões (6.9) e (6.11), resultando (para o ganho do extensómetro de 2.1):

$$\varepsilon \cong \frac{4}{\mathrm{G}} \cdot \frac{\mathrm{Vo}' - 1.65}{1000\ x\ Vi} = \frac{4}{2.1} \cdot \frac{\frac{\mathrm{ADC\ x\ 3{,}3}}{1023} - 1.65}{1000\ x\ 5} \tag{6.13}$$

A expressão (6.13), será o modelo matemático utilizado no sistema de supervisão e controlo, para calcular o valor nas unidades correctas, que serão mostrados na interface ao operador e guardados na base de dados.

6.2.2 - Acelerómetros

Os acelerómetros utilizados no sistema são 2 do tipo ADLX203, os quais estão apresentados na secção 6.1.2 e ainda os acelerómetros existentes na placa de desenvolvimento do Waspmote, os LIS3LV02DL da STMicroelectronics, que são também uns acelerómetros do tipo MEMS, com gamas de medição de -2g a +2g. Estes ultimos não necessitam de qualquer circuito de aquisição, visto todo esse circuito já estar implementado na placa de desenvolvimento, assim como o modelo matemático para obtensão dos resultados em g.

Todavia, para os acelerómetros da Analog Devices, foi necessário o desenvolvimento de um circuito de aquisição, representado na figura 6.8. Estes produzem uma saída de tensão de

0 a 5V que tem que ser ajustada para uma tensão de 0 a 3.3V. Para isso, utilizou-se um circuito (figura 6.8) referente a um divisor de tensão, implementado no integrado OPA2350.

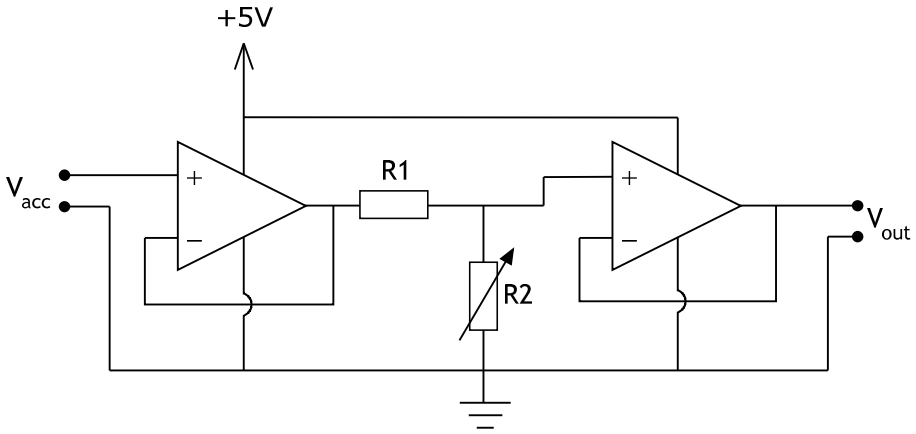


Figura 6.8 - Circuito de aquisição e condicionamento de sinal dos acelerómetros

Foram então, desenvolvidas duas placas de circuito impresso com este circuito, uma para cada um dos sensores disponíveis, ficando essas placas com o aspecto da figura 6.9. Cada uma destas placas possui dois divisores de tensão (figura 6.8), pois o acelerómetro permite a medição de acelerações bidimensionalmante, ou seja, para dois eixos. Permitindo assim, numa mesma PCB o condicionamento do sinal, referente a estes dois eixos dos acelerómetros.

Figura 6.9 - PCB dos circuitos de aquisição e condicionamento de sinal dos acelerómetros

O módelo matemático para obtenção dos valores em g é neste caso mais simples, que para os extensómetros. Esquecendo o circuito de condicionamento de sinal, sabe-se que o acelerómetro mede 1g por cada *volt* de saída que impõe e para 0g produz uma saída de 2.5V. Então, o valor máximo que produz na sua saída de tensão é 2.5+1.7=4.2V (1.7g) e o valor mínimo é de 2.5-1.7=0.8V (0g). Aplicando agora, o circuito de condicionamento de sinal que aplica um divisor à tensão de saída do acelerómetro de factor 3.3/5=0.66, percebe-se que para 1.7g o valor de tensão à entrada dos Waspmotes é de 4.2x0.66 = 2.772V; para 0g, é de 2.5x0.66=1.65; e para -1.7 é de 0.8x0.66=0.528V. O modelo matemático de calculo da aceleraçã em g, sabendo que o acelerometro mede 1g/V, está a seguir representada:

$$Acc\ (g) = \frac{\text{Vo}}{0.66} - \text{offset} = \frac{\text{ADC x 3,3}}{1023 \text{ x } 0.66} - 2{,}5 \qquad (6.14)$$

6.2.3 - Temperatura

Os sensores de temperatura disponibilizados para este projecto, eram do tipo RTDs, os quais apresentam uma grande desvantagem dado necessitarem de potência relativamente elevada para funcionarem correctamente, não sendo desta forma possível utiliza-los no sistema. Para colmatar esta lacuna, utilizou-se então o sensor de temperatura existente também na placa de desenvolvimento do Waspmote, que é um sensor de temperatura de IC, que permite medir temperaturas num intervalo de -40ºC a +85ºC.

Não foi necessário o desenvolvimento de um modelo matemático para obter o valor final de temperatura, pois esses cálculos já estão implementados numa biblioteca do API do Waspmote, obtendo-se assim directamente o valor em graus Celsius.

6.3 - Implementação da Unidade de Processo

Processo Geral

O processo geral da Unidade de Processo é em muito idêntico ao da Unidade de Comunicação e Controlo Inteligente, representado na figura 5.21. As principais diferenças assentam no facto de, neste caso não existir uma interrupção do estado de bloqueio por um temporizador e também, no facto da recepção de mensagem que permite sair desse bloqueio não ser adquirida por uma comunicação UART, mas sim por uma comunicação sem fios, XBEE.

Tratamento das Tramas de Alarmes

As Unidades de Processo (Escravos) são as responsáveis por detectar a ocorrência de alarmes ao nível da estrutura (ponte). Estas, ao receberem um pedido de leitura dos alarmes originário das Unidades de Comunicação e Controlo, realizam o processo apresentado na figura 6.10.

Assim, verificam numa primeira instância as Unidade de Processo, verificam se já possuem algum alarme que tenha sido captado numa análise anterior. Caso existam já esses alarmes, é escrito o primeiro da lista na mensagem de resposta e posteriormente, essa resposta é enviada ao seu Mestre. Se não possuir nenhum alarme na sua memória, a Unidade de Processo, realiza o processo sequencial representado na figura 6.10. O qual consiste numa leitura inicial de um valor relativo a todos os seus sensores, leituras essas que são primeiramente comparadas a um limite superior e inferior de medições, que permita verificar se os sensores estão ou não a medir correctamente, por exemplo, se um sensor que esteja a medir a temperatura ambiente se apresentar um resultado de 200ºC está obviamente com algum problema, sendo neste caso activado um alarme de sensor fora de ordem. Se o valor medido pelos sensores não tiver ultrapasso este limite, é seguidamente comparado com um valor de *threshold* superior e inferior, valor este que indicará a passagem dos valores medidos nos sensores por um limite de alarme, que pode implicar consequências graves no sistema. Desta forma, previne os responsáveis pelas obras dessa situação para que estes possam actuar sobre a estrutura da forma que acharem necessário. Assim sendo, este é provavelmente o alarme mais importante do sistema de supervisão e controlo. Depois de comparados todos os valores medidos pelos sensores aos limites enumerados, segue-se a verificação das baterias dos Escravos. Esta verificação é feita recorrendo a uma biblioteca do API do Waspmote, que fornece o valor em percentagem, da carga actual da bateria. Se esse valor da carga for menor que um limite estabelecido, no caso definiu-se 10%, é activado um alarme.

Os limites de sensor fora de ordem e *thresholds*, a que os valores medidos são comparados, foram definidos para o sistema protótipo implementado, sem nenhum estudo critério rigoroso, como valores de teste, pois, a determinação final destes limites, dependeria de caso para caso e deveria ser definida pelos engenheiros estruturais e pelos responsáveis pelas obras, isto por serem as entidades mais interessadas (*stakeholders*) no significado destes alarmes.

Os alarmes que vão sendo adquiridos em todas estas comparações, são guardados provisoriamente na memória da Unidade de Processo, assim, no fim de todas as verificações esta averigua se registou algum alarme e em caso afirmativo, envia para o Mestre o primeiro alarme que tenha verificado durante este processo, sendo os restantes enviados em pedidos posteriores. Se não se verificou nenhum alarme, é enviada ao Mestre uma resposta a indicar essa não ocorrência.

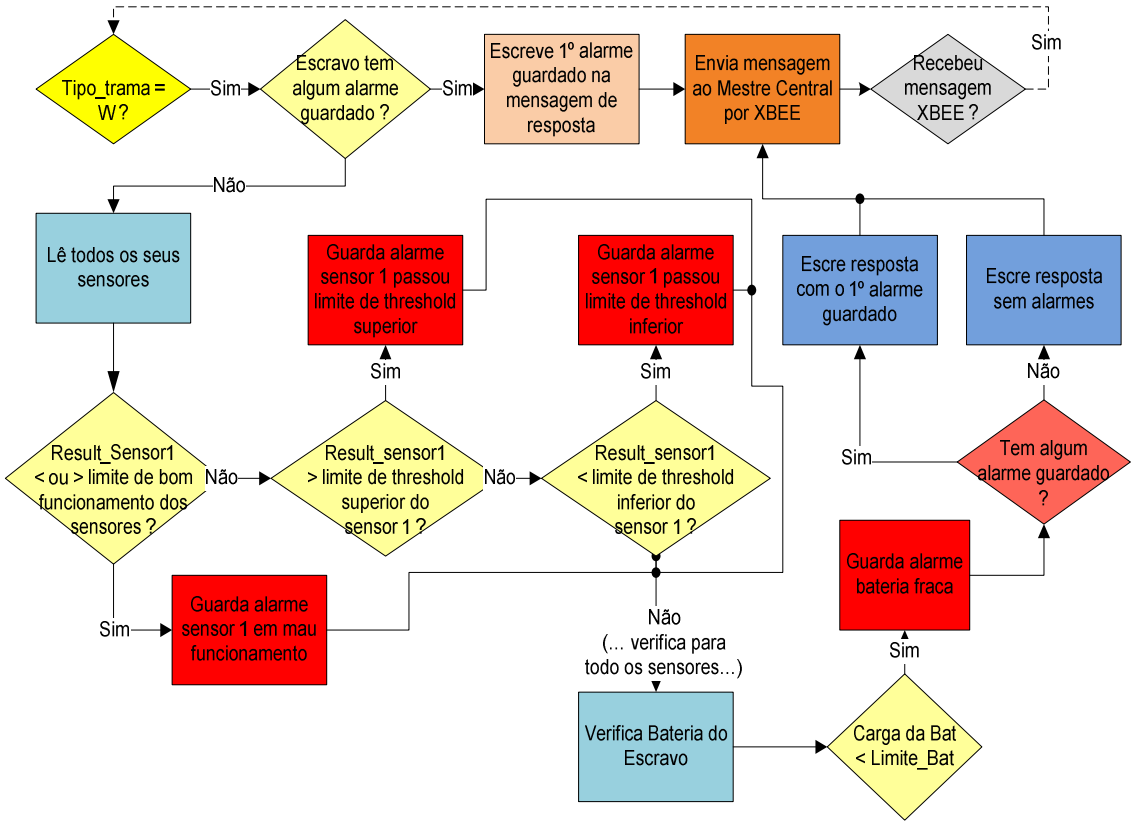


Figura 6.10 - Processamento relativo às tramas de alarmes nos diapositivos de processo

Tratamento das Tramas tipo C, R, A, M, e L

Na figura 6.11, pode verificar-se o tratamento efectuado nos Escravos relativamente às tramas: de sincronização do seu relógio interno (trama tipo C); da leitura do estado de um actuador (trama tipo A); da mudança de estado de um actuador (trama tipo M); da leitura de vários valores consecutivos (trama tipo L); e ainda, da leitura de valores dos sensores activos nos Escravos (trama tipo R).

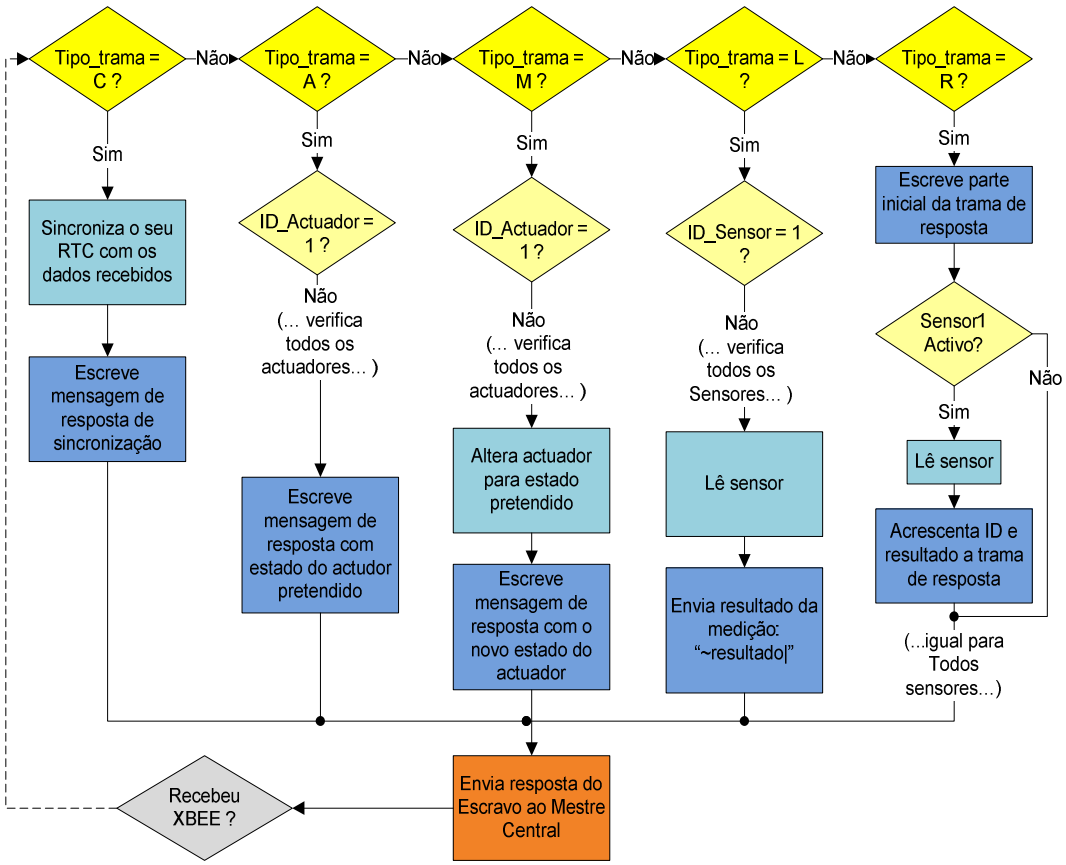


Figura 6.11 - Processo de tratamento das tramas tipo C, R, A, M e L nos dispositivos de processo

No primeiro caso, tratamento das tramas tipo C, o processo é muito simples, o qual passa por actualizar os valores do RTC de acordo com os dados recebidos na trama de sincronização e em seguida escreve a mensagem de resposta que será enviada posteriormente ao Mestre.

Quando se recebeu uma trama do tipo A, procura-se o actuador que o Mestre pretende ler e escreve-se a mensagem de resposta ao Mestre com o estado desse actuador.

No caso das tramas tipo M, é actualizado o estado do actuador pretendido, para o estado pedido pelo Mestre e é escrita a mensagem de resposta com a confirmação desse pedido.

As tramas tipo L, são tratadas neste dispositivo de uma forma muito simples. Primeiramente procura-se o sensor pretendido para a leitura, quando encontrado, esse sensor é lido e é enviada uma mensagem de resposta muito simples apenas com esse valor do género: "~1234%CRC|". Esta pequena mensagem, permite eliminar algum *overhead* nas mensagens trocadas, para que a aquisição destes valores seja possível a uma frequência elevada por parte dos Mestres.

Por fim a mensagem de leitura dos sensores activos (tipo R) é tratada da forma ilustrada na figura. Onde, é escrita a primeira parte da mensagem de resposta respectiva, sendo que em seguida se procuram quais os sensores activos, através da mensagem de definição da função de leitura e activação dos sensores, e desses sensores são lidos os valores que estão a medir nesse instante, sendo esses resultados acrescentados ordenadamente na mensagem de resposta, precedidos pelo seu ID do sensor. Quando todos os sensores são verificados, é enviada a trama de resposta à Unidade Superior.

Trama de Definição da Função de Leitura dos Escravos e Activação dos seus Sensores

Por último, falta tratar a trama tipo F, que permite ao operador definir na interface a função de leitura para os Escravos, bem como, os sensores que se pretendem ler como essa função. O processamento referente a este tipo de trama está representado na figura seguinte (figura 6.12).

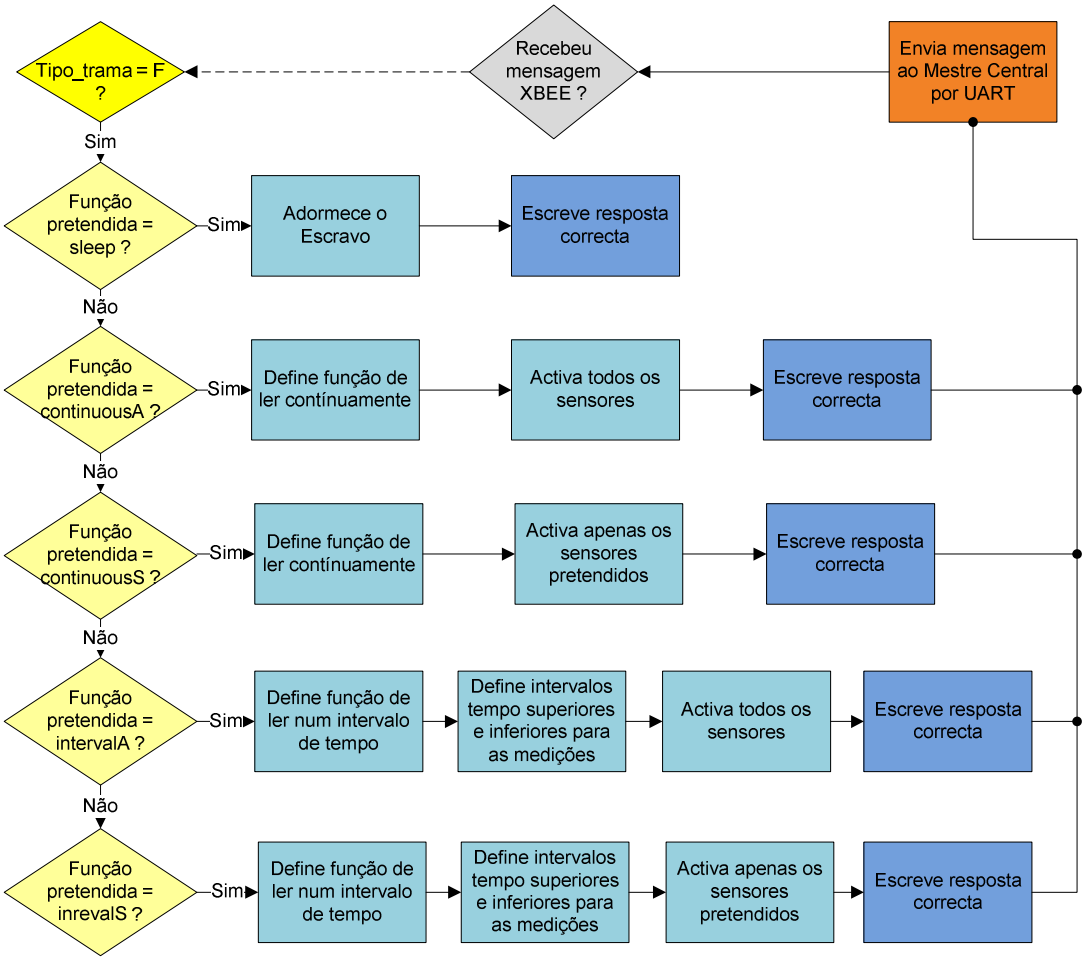


Figura 6.12 - Processo de tratamento das tramas de definição do modo de leitura e activação de sensores dos dispositivos de processo

Do ponto de vista do dispositivo de processo, a definição da função de leitura em modo contínuo ou num intervalo de tempo, não tem qualquer significado, pois essas tarefas são neste sistema tratadas na interface. No entanto, com esta característica implementada poderia ser desenvolvidas futuramente novas funcionalidades a estes dispositivos, como medir autonomamente os seus sensores e guardar esses dados numa base de dados local, por exemplo, num cartão SD (*Secure Digital*) e envia-los posteriormente às unidades superiores na sua totalidade, ou sob a forma de informação qualitativa.

Capítulo 7

Resultados e sua Discussão

No presente capítulo serão apresentados e discutidos os resultados adquiridos do sistema protótipo implementado (figura 7.1), os quais permitem caracterizar o mesmo. Estes resultados são divididos em duas partes principais: os resultados dos sistema de supervisão e controlo, relativos aos tempos de atrasos das comunicações (RS-232, CAN e *wireless*) e do processamento; e ainda os resultados obtidos de ensaios de medições dos sensores utilizados, ensaios estes que foram realizados no LABEST – Laboratório da Tecnologia do Betão e do Comportamento Estrutural, da Faculdade de Engenharia.

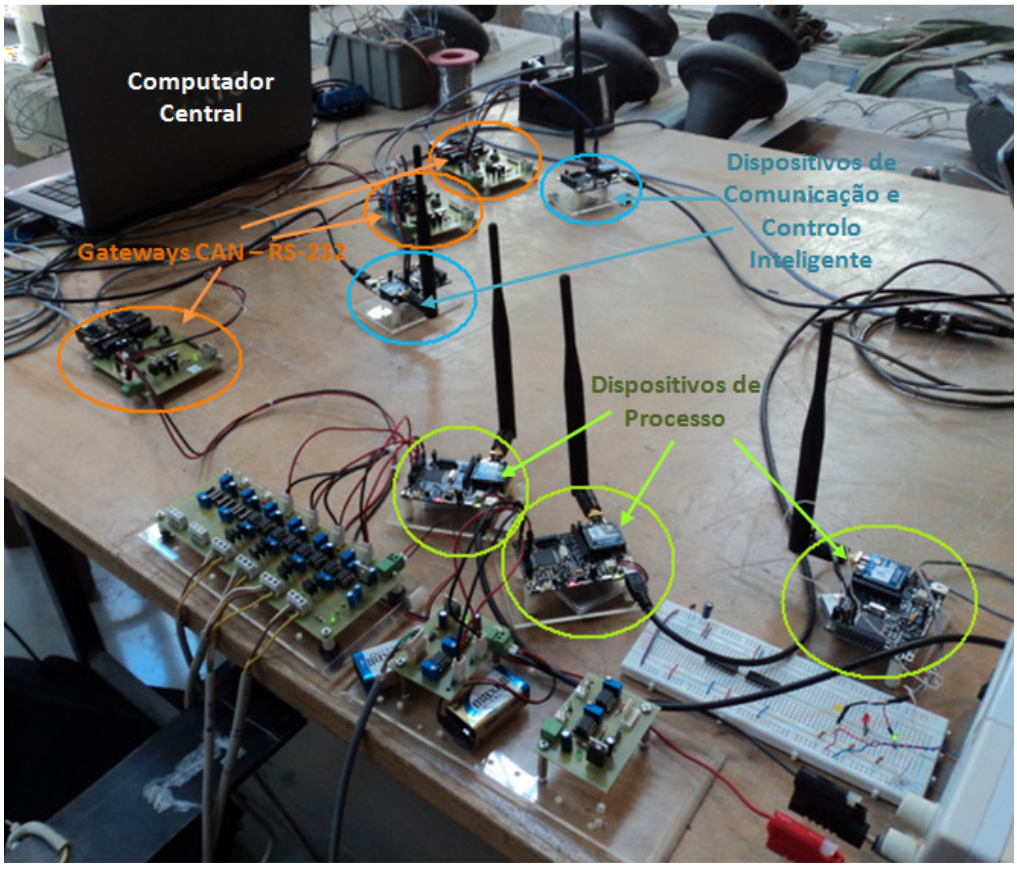


Figura 7.1 - Protótipo do Sistema Implementado

7.1 - Velocidade das Comunicações

Foram realizados vários tipos de testes da velocidade às diferentes comunicações existentes no sistema, para perceber os atrasos do envio da informação no sistema implementado. Para isso, realizaram-se ensaios individuais, a cada tipo de comunicação e ainda, ensaios de todas as comunicações em conjunto.

Todos estes ensaios foram realizados na óptica do sistema, isto é, utilizando as tramas relativas aos protocolos de comunicação definidos para este sistema que são sujeitas aos processos de tratamento de mensagens nos diferentes componentes referentes a cada uma.

7.1.1 - Comunicações RS-232

O primeiro teste realizado diz respeito ao tempo necessário para as comunicações RS-232, utilizadas na comunicação entre as *gateways* CAN – RS-232 (AT90CAN64) e os Waspmotes, bem como, entre as *gateways* e a interface HMI. Assim, será medido o atraso desta comunicação, desde que a trama é enviada até que esta seja completamente recebida pelo Waspmote, isto implica a recepção da trama e todo o processamento necessário para a sua verificação (figura 5.15). O procedimento experimental utilizado na medição do atraso das comunicações RS-232 (ou UART neste caso) entre a *gateway* e o Waspmote, foi o seguinte:

1. Enviar repetidamente, a cada 100ms uma trama aleatória, do AT90CAN64 para o Waspmote;
2. Activar, durante 1ms, uma das saídas do AT90CAN imediatamente após o envio da trama;
3. Activar, durante 1ms, uma das saídas do Waspmote depois da trama ser recebida e verificada;
4. Conectar a cada uma das saídas, referidas em 2 e 3, uma ponta de prova diferente do mesmo osciloscópio;
5. Verificar a diferença de tempo entre as duas saídas.

O resultado obtido deste ensaio pode ser verificado na seguinte ilustração (figura 7.2).

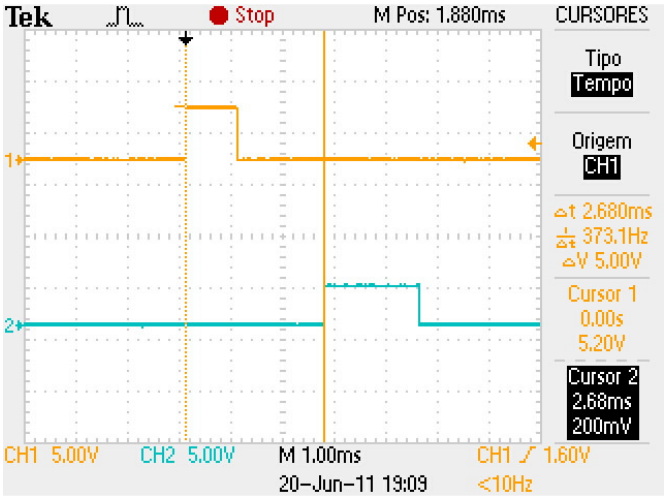


Figura 7.2 – Resultado do teste de velocidade RS-232

Repare-se que a escala de tempo utilizada no osciloscópio é de 1 ms/div, sendo que, conectado ao canal 1 do osciloscópio estava a saída da *gateway* e ao canal 2 a saída do Waspmote.

Verifica-se então na figura anterior que a comunicação RS-232 impõe um atraso total no sistema implementado de 2.68ms.

A trama enviada foi "~1#1#8#M#2#2%204|\r\n", ou seja, 19 bytes => 19 x 8 = 152 bits, a estes deve ainda ser adicionado o *start* e o *stop* bit utilizado nas tramas UART para envia um byte (mais 2 bits por byte enviado), então 152 + 19 x 2 = 190 bits, que a 115200 bps, deveria demorar cerca 190/115200 = 1.65ms a ser transmitidos. O valor obtido e o valor esperado são de muito próximos, a pequena diferença entre estes, pode ser justificada ao considerar o tempo de processamento necessário ao Waspmote para receber e verificar a trama.

7.1.2 - Comunicações CAN

O seguinte teste diz respeito à velocidade de comunicação CAN entre as gateways do sistema. Sendo que, estas gateways estão definidas para comunicar com uma largura de banda de 100 kbits/s.

O procedimento experimental, realizado para fazer este teste de velocidade foi o seguinte:

1. Enviar repetidamente, a cada 100ms uma trama CAN *standard* aleatória, de uma das gateways CAN – RS-232 (AT90CAN64) para outra;
2. Activar, durante 1ms, uma das saídas do AT90CAN transmissor, imediatamente após o envio trama;
3. Activar, durante 1ms, uma das saídas do AT90CAN64 receptor imediatamente após a recepção da trama;
4. Conectar a cada uma das saídas, referidas em 2 e 3, uma ponta de prova diferente de um mesmo osciloscópio;
5. Verificar a diferença de tempo entre as duas saídas.

Na figura 7.2, está representado o resultado obtido, com o osciloscópio numa escala temporal de 5ms/div e com a *gateway* emissora representada no canal 1 e a receptora do canal 2.

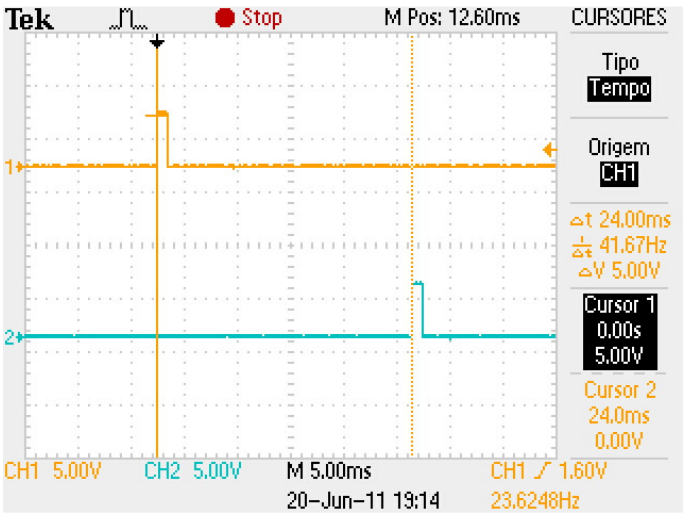


Figura 7.3 - Resultado do teste de velocidade da comunicação CAN

Verifica-se então, que o atraso de comunicação entre as *gateways* CAN é de aproximadamente 24ms. Este resultado, está longe de ser próximo do tempo de transmissão de uma trama CAN *standard* (cerca de 111 bits) a 100 kbit/s, que seria de aproximadamente 1,11ms, no entanto, isto pode ser explicado recordando todo o processamento e processos realizados pelos dispositivos CAN, referentes à camada de lógica/ligação do modelo OSI, como por exemplo, os processos de detecção de erros (secção 5.2.1). Este processamento no caso da comunicação por RS-232 não existe, visto esta apenas implementar a camada física do modelo OSI. Também, todo o processamento necessário quer para o envio, quer para a recepção referentes às bibliotecas da comunicação CAN da AT90CAN64, é um factor que pode influenciar bastante neste atraso da comunicação.

7.1.3 - Comunicações *Wireless*

Um outro atraso de comunicação de elevada relevância para o sistema é o atraso imposto pelas comunicações *wireless* entre os Waspmotes. Sendo este, outro dos testes realizados para caracterizar o funcionamento do sistema.

O procedimento experimental deste ensaio foi o seguinte:

1. Enviar repetidamente, a cada 100ms uma trama aleatória, de um dos Waspmotes para outro;
2. Activar, durante 1ms, uma das saídas do Waspmote transmissor, imediatamente após o envio trama;
3. Activar, durante 1ms, uma das saídas do Waspmote receptor após a recepção completa da trama;
4. Conectar a cada uma das saídas, referidas em 2 e 3, uma ponta de prova diferente do mesmo osciloscópio;
5. Realizar uma medição para a distância de cerca de 1m entre os dispositivos;
6. Realizar uma outra medição para a distância de cerca de 2m entre os dispositivos.

Os resultados obtidos para as duas distâncias, sendo 2m praticamente o máximo que se consegue utilizando as ponta de prova de um mesmo osciloscópio, são os mesmos e estão representados na figura 7.4, onde o osciloscópio está numa resolução de tempo de 5ms/div.

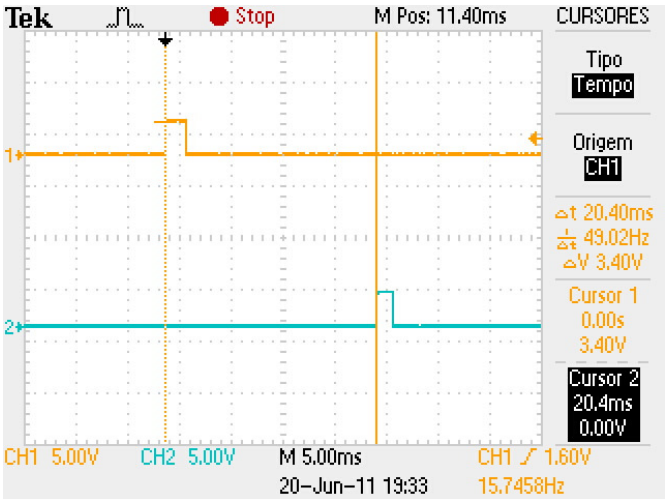


Figura 7.4 - Resultado do teste de velocidade da comunicação *wireless*

Apura-se então, que para o envio da trama "~1#1#8#M#2#2%204|", 17 bytes, mais 11 bytes fixos de elementos de controlo das tramas 802.25.4, mais 20 bytes (considerando que se utiliza o máximo) de dados de identificação da origem e destino e ainda, mais 11bytes de *acknowledge* do destinatário, dando um total de 17+11+20+11=59 bytes para esta comunicação é imposto um atraso de 20.4ms, que está também bastante longe do valor esperado para a velocidade de comunicação utilizada, de 59x8 bits/250kbps = 1,88 ms. A justificação para esta diferença pode ser, tal como para o caso anterior, devida ao tempo de processamento necessário para executar todas as tarefas das bibliotecas utilizadas da API do Waspmote, para a realização desta comunicação. Existindo também neste caso para além dessas tarefas, os processos de verificação da trama recebida implementados.

Distância Máxima de Comunicação *Wireless*

Ainda relativamente às comunicações sem fios, foram realizados alguns testes referentes à distância máxima a que se consegue comunicar. Segundo dados do fabricante com estes dispositivos é possível comunicar até 500 em espaço aberto, no entanto, nos testes realizados nas instalações da Faculdade de Engenharia da Universidade do Porto apenas se conseguiu comunicar a uma distância máxima de cerca de 160m, como se pode verificar na figura 7.5. Esta discrepância pode ser devida à pouca potência energética das baterias dos Waspmotes, visto que, a potência necessária para transmitir dados recorrendo aos módulos XBEE é proporcional à distância a que estes se encontram. Uma outra justificação pode assentar na existência de algumas agressões externas, provocadas por exemplo, pelas ondas de rede de internet *wireless* existentes nestas instalações.

Figura 7.5 - Distância máxima das comunicações *wireless*

7.1.4 - Comunicações Globais

Por último, e talvez mais importante, realizaram-se testes relativos aos tempos de demora desde que as mensagens são enviadas até serem recebidas, de e pela interface, ou seja, o atraso imposto pelo sistema desde que a mensagem é envia pela interface, passando primeiramente pela comunicação RS-232 até às gateways CAN – RS-232, em seguida pelas comunicações CAN das gateways, novamente por RS-232 para os Waspmotes e finalmente, por

wireless entre os Waspmotes (caso se aplique) das Unidades de Comunicação e Controlo Inteligente, até às Unidades de Processo.

Dos testes anteriores, verificou-se que: para as comunicações RS-232 o atraso obtido era de 2.68ms, o qual deve ser considerado duas vezes (interface gateway e gateway Mote), ou seja, aproximadamente 2.68x2=5.36ms; para as comunicações CAN o atraso era de 24ms; e para as comunicações wireless de 20.4ms. Da soma de todos estes tempos obtém-se 52.44ms. Então conjuntamente, ou seja, na comunicação num sentido mais a comunicação no outro, obtém-se 52.44x2=104.88ms.

Nestes ensaios, é importante perceber o tempo de comunicação respectivo para cada um dos tipos de trama, visto utilizarem um número de bits (no caso do RS-232 e *wireless*) e um número de tramas CAN (para as comunicações CAN) algo distinto entre elas.

O procedimento experimental utilizado para a realização destes ensaios foi o seguinte:

1. Enviar uma trama (de cada um dos tipos possíveis) desde a interface;
2. Iniciar um temporizador na interface (com resolução aos milissegundos) imediatamente após o envio da trama;
3. Parar o temporizador no instante em que se recebe a resposta;
4. Guardar o atraso de tempo verificado;
5. Realizar 5 vezes o processo.

Tramas de Alarmes - Tipo W

Os resultados obtidos para as tramas de alarmes estão representados na tabela 7.1.

Tabela 7.1 — Atrasos das comunicações conjuntas para as tramas de alarmes

Número do Ensaio	Atraso (ms)
Ensaio 1	437
Ensaio 2	437
Ensaio 3	453
Ensaio 4	432
Ensaio 5	453
Média	442.4

Nas tramas de alarmes, a interface (Unidade de Interface Comando e Controlo) envia uma mensagem para leitura de alarmes a cada um dos seus Mestres (que não contêm ainda nenhum alarme na memória), os quais têm que perguntar aos seus Escravos. Para teste utilizou-se apenas o Mestre 10 que só contém associado o Escravo 10.

Sabe-se então, que das comunicações globais poderia-se obter um valor aproximado de 105ms, contudo, as discrepâncias verificadas nestes resultados devem-se a todo o processamento para esta tarefa, relativo aos processos explicados nos capítulos anteriores. Ainda neste caso, sabe-se que a mensagem é enviada para os dois Mestres (apesar de no teste apenas se receber de um deles) o que também impõe atrasos que podem ser significativos. No entanto, relativamente ao tempo mínimo (tempo-real) imposto para este sistema, que era de cerca de 1s, são resultados muito favoráveis.

Trama de Sincronização dos Relógios – Tipo C

No que respeita aos ensaios das tramas de sincronização dos relógios, obtiveram-se os resultados da tabela 7.2.

Como era de esperar, os atrasos das comunicações destas tramas são maiores que no caso anterior, pois o número de bytes utilizados para as tramas do protocolo geral é bem maior e utilizam, comparativamente às anteriores, três em vez de uma trama CAN. E tal como as anteriores, também neste caso é enviada uma mensagem a cada um dos dispositivos de comunicação e controlo inteligente.

Tabela 7.2 — Atrasos das comunicações conjuntas para as tramas de sincronização dos relógios internos dos Waspmotes

Número do Ensaio	Atraso (ms)
Ensaio 1	625
Ensaio 2	688
Ensaio 3	672
Ensaio 4	687
Ensaio 5	672
Média	668.8

Trama de Definição das Funções dos Escravos – Tipo F

Para estes ensaios, utilizou-se uma trama com a função *intervalS*, por ser o tipo de trama que ocupa mais bytes no protocolo geral. Obtiveram-se então os resultados apresentados na tabela 7.3.

Tabela 7.3 — Atrasos das comunicações conjuntas para as tramas de definição das funções dos Escravos

Número do Ensaio	Atraso (ms)
Ensaio 1	1094
Ensaio 2	1094
Ensaio 3	1109
Ensaio 4	1094
Ensaio 5	1125
Média	1103.2

Por ser este, o tipo de trama onde se utiliza o máximo número de bytes e tramas CAN (excluindo as trama tipo L), era de esperar que esta fosse a comunicação mais demorada. O que de facto se verifica, ultrapassando este atraso até 1s. O que não é no entanto crítico, pois não se espera que esta mensagem seja utilizada continuamente, logo não ocupará a rede

muitas vezes. E não faz qualquer diferença para um utilizador, alguns milissegundos a mais para além de um segundo (tempo-real definido para o sistema).

Trama de Leitura dos Sensores dos Escravos – Tipo R

Os resultados destas tramas são de alguma relevância, por serem estas as tramas utilizadas sempre que se pretende aceder aos valores medidos pelos sensores e que, ao contrário das tramas tipo F, espera-se que sejam enviadas com muita frequência. Podendo ser mesmo, a par das tramas de alarmes, das tramas mais repetidas no sistema.

Os resultados obtidos para este tipo de trama foram os da tabela 7.4. Estes são resultados muito positivos para este tipo de trama, pois pode-se constatar que em média demoraram apenas cerca de 336 ms, o que é um valor menor que metade do tempo real imposto. Importa destacar que, para este tipo de trama o pedido efectuado pelo Mestre Central é muito pequeno, mas a resposta já tem um tamanho assinalável, ocupando até duas tramas CAN. E para além disso, o processamento envolve a leitura dos valores de entrada dos ADCs, que como ficará em seguida evidente é mais rápida que o processamento da leitura das portas E/S.

Tabela 7.4 — Atrasos das comunicações conjuntas para as tramas de leitura dos sensores dos Escravos

Número do Ensaio	Atraso (ms)
Ensaio 1	313
Ensaio 2	343
Ensaio 3	328
Ensaio 4	344
Ensaio 5	351
Média	335.8

Trama de Leitura do Estado dos Actuadores dos Escravos – Tipo A

Obtiveram-se neste caso os resultados da tabela 7.5, apresentada em baixo.

Tabela 7.5 — Atrasos das comunicações conjuntas para as tramas de leitura do estado dos actuadores dos Escravos

Número do Ensaio	Atraso (ms)
Ensaio 1	328
Ensaio 2	332
Ensaio 3	330
Ensaio 4	343
Ensaio 5	344
Média	335.4

Como se verificar, estes ensaios apresentam praticamente os mesmos resultados que os da trama tipo R, no entanto, à que considerar que neste caso são utilizados quer no pedido quer na resposta um número de bits muito reduzido e apenas uma trama CAN. Podendo assim concluir-se que a leitura do estado de uma entrada ou saída do Waspmote é mais lenta do que a leitura do valor do ADC, o que pode ter a ver com as bibliotecas da API do componente.

Trama de Mudança de Estado dos Actuadores dos Escravos – Tipo M

A trama seguinte, diz respeito à trama de mudança do estado dos actuadores, sendo que neste caso, ao contrário da anterior, a trama de resposta recebida na interface espera-se que seja exactamente a mesma que a trama enviada, pois são utilizados neste caso mais bytes que no caso anterior (no protocolo geral). Para além disso, as acções efectuadas pelo Mestre não são apenas ao nível de software, como para as tramas tipo A, aqui é necessária uma manipulação também do hardware, por se estar a alterar uma saída analógica do Waspmote. Esperando-se por isso, que os resultados sejam ligeiramente superiores ao caso anterior. Essa previsão está assim confirmada na tabela 7.6, que apresenta os resultados dos testes obtidos para este tipo de trama.

Tabela 7.6 — Atrasos das comunicações conjuntas para as tramas de mudança de estado dos actuadores dos Escravos

Número do Ensaio	Atraso (ms)
Ensaio 1	375
Ensaio 2	343
Ensaio 3	375
Ensaio 4	344
Ensaio 5	348
Média	357

Trama de Leitura de Vários Valores dos Sensores dos Escravos – Tipo L

Por fim, faltam analisar os resultados das tramas tipo L, como já foi realçado, esta trama impõe uma comunicação e um processamento bastante complexo a todos os elementos do sistema. Os testes foram realizados para a leitura de 200 valores em 10 tramas do protocolo geral, o que implica 50 tramas CAN. Para além disso, ainda fazem parte desta comunicação as "tramas de controlo" (tramas de índice 1, índice 2 e índice 3). Portanto, o atraso esperado para a comunicação de todos estes valores espera-se que seja bastante significativo e está representado na tabela 7.7.

São como se pode verificar na tabela, de facto os resultados são muito elevados, andando na ordem dos 22 segundos. Portanto, é necessário muito cuidado na utilização destas tramas, porque esta trama impede que a rede não seja acedida durante um intervalo de tempo bastante significativo, o que não permite que as mensagens prioritárias (alarmes) sejam acedidas durante este atraso de tempo.

Tabela 7.7 — Atrasos das comunicações conjuntas para as tramas de leitura de vários valores dos sensores dos Escravos

Número do Ensaio	Atraso (ms)
Ensaio 1	21907
Ensaio 2	22094
Ensaio 3	21750
Ensaio 4	22437
Ensaio 5	21875
Média	22012.6

7.2 - Ensaios de Medições dos Sensores

Foram realizados alguns ensaios no laboratório de estruturas (LABEST), do departamento de Engenharia Civil da FEUP. Onde se utilizou uma pequena barra metálica, com 74 cm de comprimento, 8 cm de largura e 2.5 cm de espessura, como objecto de teste. Mas repare-se que na zona dos extensómetros ES1 e EI1, a barra tem uma secção mais pequena. A mesma contém 6 extensómetros colados na sua superfície, três por cima e três por baixo e ainda, um acelerómetro colado na parte superior da barra que permitirá verificar as vibrações da ponte. Esta barra pode assim ser observada na fotografia da seguinte figura (figura 7.6).

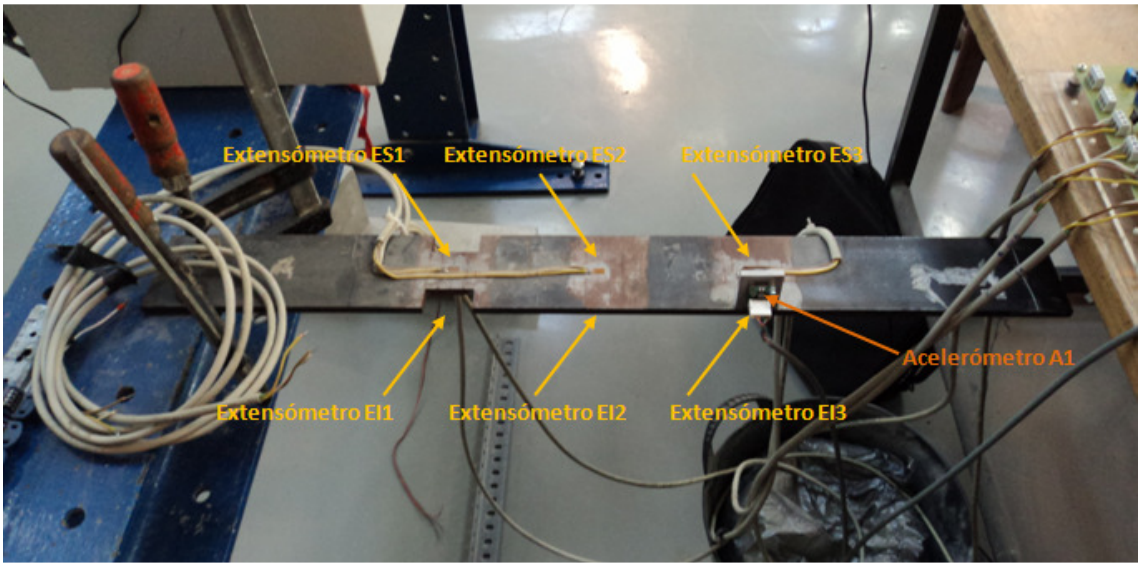


Figura 7.6 - Barra metálica para ensaios dos sensores

Esta barra encontrava-se já com os extensómetro colados (figura 7.7a), pelo que foi o material disponibilizado para validação e teste do nosso sistema, sendo apenas necessário colar-lhe depois o acelerómetro. O outro acelerómetro que se esperava utilizar nestes ensaios, encontrava-se colado a um outro objecto com uma cola muito forte, pelo que não se utilizou. Também, os acelerómetros dos Waspmotes não foram utilizados, pois ao acopla-los a

esta barra podia danificar-se este componente, sendo então, apenas utilizado para estes ensaios um acelerómetro (figura 7.7b). Para além disso e infelizmente, dois dos extensómetros da barra, o superior e o inferior do meio da barra (ES2 e EI2), encontravam-se em muito mau estado, produzindo por isso resultados sem qualquer critério não sendo deste modo utilizados. Assim, utilizaram-se nestes ensaios, apenas 4 extensómetros, 1 acelerómetro e os 2 sensores de temperatura do Waspmotes, que apesar de não ser o proposto, pois tinha-se pensado inicialmente em 6 extensómetro, 4 acelerómetros e 2 sensores de temperatura, são suficientes para a validação do sistema de supervisão e controlo.

Para realizar os ensaios, que serão em seguida apresentados, colocou-se na extremidade solta da barra, um peso de 50kg que provocou uma deformação na mesma, permitindo deste modo, verificar o funcionamento dos extensómetros e até mesmo dos acelerómetros, pois, com uma pequena força aplicada à barra com o peso, é provocada uma vibração da mesma durante um pequeno intervalo de tempo.

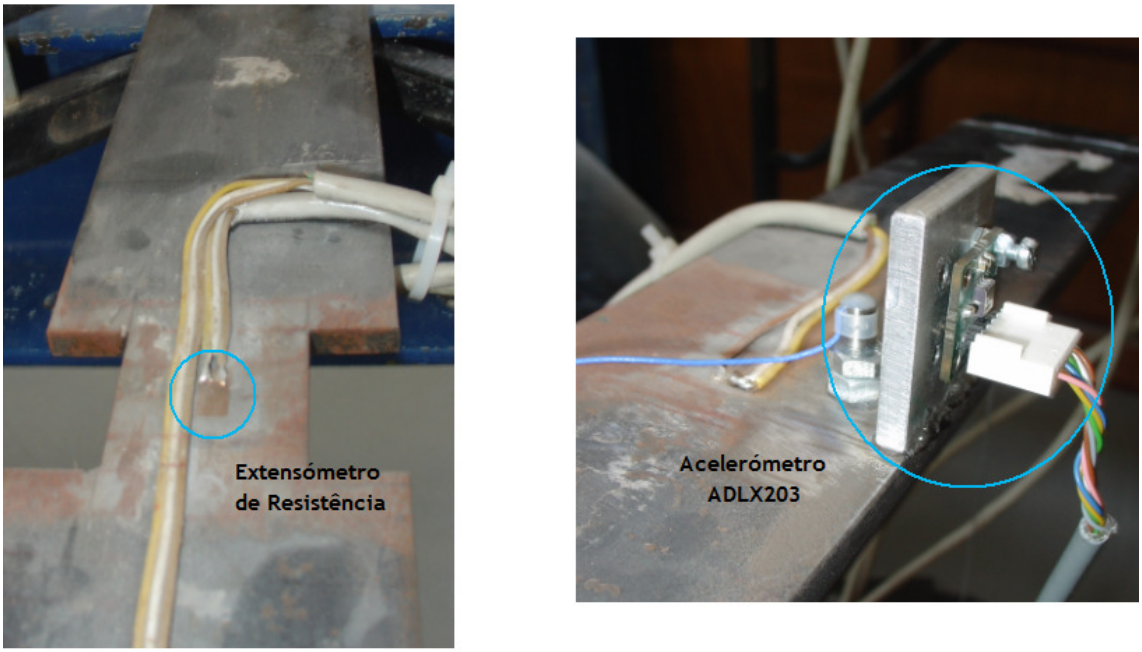


Figura 7.7 – a) Extensómetros e b) Acelerómetro - barra metálica de teste

7.2.1 - Ensaios aos Acelerómetros

Realizaram-se dois ensaios para teste dos acelerómetros, com o seguinte procedimento experimental:

1. Colocou-se um peso de 50kg na extremidade da barra;
2. Com o sistema protótipo todo montado e ligado, enviou-se da interface, um pedido de leitura de vários valores consecutivos (trama tipo L) do acelerómetro específico;
3. Aplicou-se uma pequena força (manual) na extremidade da barra provocando a sua oscilação durante um curto espaço de tempo, imediatamente a seguir ao pedido de leitura da trama L (Ensaio 1);
4. Os 200 dados recebidos, posteriormente, na interface são automaticamente guardados na base de dados e é desenhado na interface o gráfico relativo ao teste a esse teste;

5. Realizou-se um novo teste (repetiu-se 2, 3* e 4), mas desta vez, não foi aplicada uma força manual à barra, mas sim, foi-lhe retirado bruscamente o peso o que provocou uma oscilação de grande intensidade na mesma durante algum tempo (Ensaio 2);
6. Verificaram-se e analisaram-se os resultados obtidos.

Nos seguintes gráficos, figura 7.8 e figura 7.9, estão representados os resultados obtidos no Ensaio 1 e Ensaio 2, respectivamente.

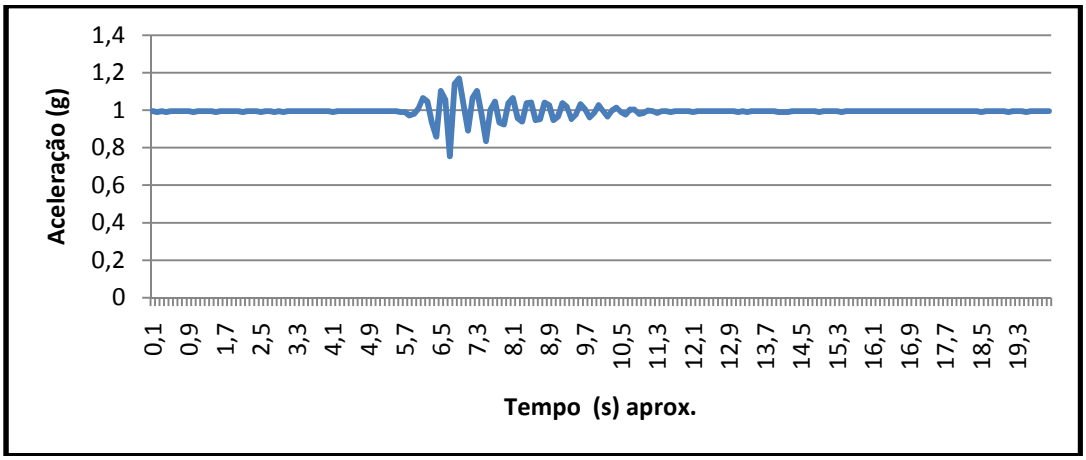


Figura 7.8 – Resultados do Ensaio 1 aos Acelerómetros

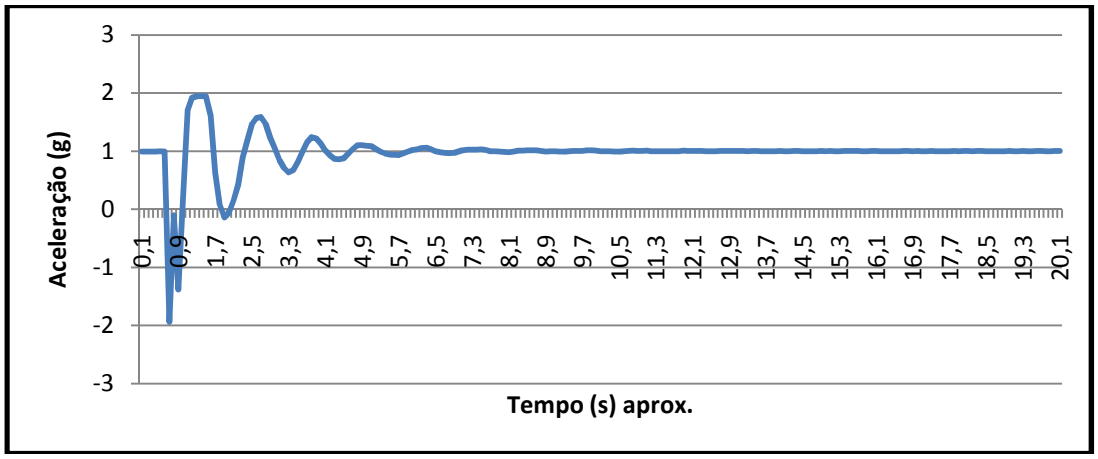


Figura 7.9 – Resultados do Ensaio 2 aos Acelerómetros

Estes gráficos, desenhados a partir dos dados recebidos na interface, podem ser comparados com os gráficos desenhados na interface figura 4.15.

Verifica-se então no ensaio 1, com dados adquiridos a cerca de 10Hz, que em repouso a aceleração medida pelo extensómetro é de 1g, ou seja, é a gravidade da terra como era de esperar. Com a força que lhe foi aplicada observa-se vibração de amplitude muito pequena, com picos em torno de 1.2g e 0.8g.

Já no ensaio 2, a vibração é muito mais acentuada, aliás como era de esperar, notando-se até que o acelerómetro atinge o seu limite de medição de acelerações positivas, que na folha de características refere que seria de ±1.7g, no entanto, conseguiram-se medições até ±2g, saturando nesse instante.

7.2.2 - Ensaios aos Extensómetros

Foram realizados dois tipos de testes aos extensómetros: o primeiro diz respeito à medição em osciloscópio dos sinais destes acelerómetros; e o segundo teste é referente a medições realizadas na interface pedindo os valores medidos e calculando esse valor em *strain*.

Ensaio 1

O procedimento experimental para o primeiro ensaio é, então, o seguinte:

1. Colocou-se, à entrada dos ADCs dos Waspmotes, aos quais estão conectados os extensómetros (intermediados pelos circuitos de condicionamento de sinal), 4 pontas de prova de um mesmo osciloscópio;
2. Aplicou-se uma pequena força manual na barra livre (sem o peso de 50kg);
3. Registou-se e verificou-se o resultado obtido.

Este resultado está representado na seguinte ilustração (figura 7.10).

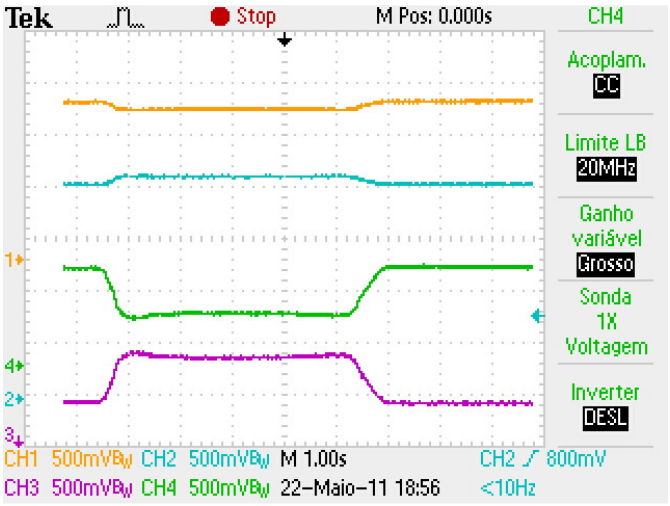


Figura 7.10 – Resultados do Ensaio 1 aos Extensómetros

Na figura anterior, a laranja está ilustrado o sinal medido no extensómetro ES3, a azul está representado o sinal do extensómetro EI3, a o sinal do ES1 e a roxo o sinal do EI1.

Como era de esperar, nos extensómetros colados na zona da barra em que a secção é mais pequena (ES1 e EI1), a extensão é muito mais acentuada que na zona perto da extremidade que quase não sofre qualquer deformação. Portanto, por esta análise concluí-se que os extensómetros estão, visivelmente, a realizar medições correctas.

Ensaio 2

No segundo ensaio, pretende-se quantificar os valores medidos pelos extensómetros. Para isso realizou-se o seguinte procedimento experimental:

1. Colocou-se um peso de 50kg na extremidade da barra;
2. Com o sistema protótipo todo montado e ligado, enviou-se da interface, um pedido de leitura dos valores dos 4 extensómetros;
3. Os dados recebidos, posteriormente, na interface são automaticamente guardados na base de dados e são numericamente demonstrados ao operador;

4. Verificaram-se e analisaram-se os resultados obtidos, calculando-se com estes (nas unidades de *strains*), o valor em *ohms* da resistência do extensómetro para cada medição (utilizando as equações descritas na secção 6.2.1.

Na tabela 7.8, estão representadas 7 medições relativas a cada um dos extensómetros.

Tabela 7.8 — Resultados do Ensaio 2 aos Extensómetros

Extensómetros ES3		Extensómetro ES1		Extensómetro EI3		Extensómetro EI1	
uStrain	R_ext	uStrain	R_ext	uStrain	R_ext	uStrain	R_ext
18,11	350,013	296,67	350,218	-79,59	349,942	-144,86	349,894
20,02	350,015	294,76	350,217	-69,11	349,949	-143,91	349,894
18,59	350,014	297,15	350,218	-60,51	349.956	-142,96	349,895
18,59	350,014	296,67	350,218	-68,63	349,950	-143,43	349,895
13,35	350,010	289,99	350,213	-66,04	349,951	-143,43	349,895

Também através destes resultados, se verifica o bom funcionamento de todos os extensómetros. Dos mesmos também se pode inferir que há uma deformação mais assinalável para os extensómetros ES1 e EI1, sendo ainda maior no caso do extensómetro superior, relativamente aos extensómetros ES3 e EI3. Isto porque, na secção mais pequena, a concentração de esforços é maior o que implica necessariamente também níveis de deformação acrescidos.

As resistências dos extensómetros apresentam variações muito pequenas, como era de esperar pela informação fornecida na folha de características das mesmas. Dado isto, assume-se então, que os valores medidos pelos extensómetros são coerentes.

7.2.3 - Ensaios aos Sensores de Temperatura

No sistema implementado, utilizaram-se apenas os sensores de temperatura integrados nos Waspmotes, então, não houve necessidade de desenvolver nenhum circuito de condicionamento de sinal ou modelo matemático, pelo que os resultados obtidos nestas medições, são lidos directamente dos Waspmotes (utilizando uma biblioteca especifica) e comparados com um termómetro de temperatura vulgar são coincidentes.

Realizou-se assim, alguns pequenos ensaios onde se pediu, pela interface, o valor de temperatura às Unidades de Processo, que responderam ambas com o valor de 27ºC (pois estas apenas apresentam resolução às unidades). Este valor quando comparado ao valor de um termómetro digital muito simples, que media 26.6ºC, é condizente, logo pode assumir-se como correctas as medições realizadas pelo sensor de temperatura utilizado.

Capítulo 8

Considerações Finais e Desenvolvimento Futuro

8.1 - Conclusões

Enquadrado pela necessidade de supervisionar e controlar o comportamento de estruturas de engenharia civil, a dissertação que agora termina constitui, assim, uma contribuição para o conhecimento das potencialidades de diversas tecnologias passíveis de se utilizarem na realização dessas tarefas de supervisão e controlo. Tecnologias estas, que para além de serem profundamente estudadas, foram ainda testadas num sistema protótipo de supervisão e controlo de estruturas, no sentido de se compreender as suas principais qualidades e adversidades, como soluções para a monitorização e controlo de processos, sendo neste caso estes processos as estruturas, ou mais especificamente pontes de Engenharia Civil.

Assim, no inicio deste documento surge a necessidade de se realizar um estudo teórico sobre o sistema de supervisão e controlo em geral, que são comummente aplicados para sistemas de automação industriais. Deste estudo fez também parte, uma análise das arquitecturas para estes sistemas de automação (centralizada, descentralizada e distribuída), com ênfase à arquitectura distribuída, que surge como a melhor arquitectura para aplicações deste género, e ainda, uma pequena apreciação da utilização da noção de tempo real, para sistemas de automação em geral. Com este estudo procurou depreender-se, o possível enquadramento de sistemas de supervisão e controlo em estruturas. Por conseguinte, visto que os sistemas SCADA são actualmente parte integrante de qualquer sistema de supervisão e controlo, foi importante explora-los e perceber a sua evolução histórica, e ainda, os elementos que os constituem: HMIs, MTUs e RTUs, que servirão como base para os capítulos posteriores.

Fundamentado este estudo, é no capítulo 3 apresentada a arquitectura e o conjunto das tecnologias utilizadas para o sistema de supervisão e controlo proposto, o qual satisfaz um conjunto algo extenso de requisitos impostos para o projecto. Esta arquitectura é composta em três níveis hierárquicos: Unidade de Interface, Comando e Controlo (nível superior); Unidade de Comunicação e Controlo Inteligente (nível intermédio); e Unidade de Processo (nível inferior). Sendo distribuído todo o processamento necessário para a implementação deste sistema por essas Unidades, que podem ser equiparadas a uma arquitectura SCADA, ao

considerar-se que as tarefas da interface entre o sistema e um possível operador, função do elemento HMI do sistema SCADA, são realizadas na Unidade Superior. Para além disso, o processamento, comando e controlo, tarefas que do ponto de vista do sistema SCADA devem ser realizadas pelos MTUs, são, no sistema proposto e implementado, distribuídas pela Unidade de Interface, Comando e Controlo e pela Unidade de Comunicação e Controlo Inteligente. E finalmente, as tarefas dos RTUs do SCADA, de medição e actuação do estado no processo, são realizadas na Unidade de Processo, sendo esta a Unidade de mais baixo nível do sistema.

No capítulo 4, procurou-se apresentar as soluções escolhidas para o desenvolvimento da Unidade de Interface, Comando e Controlo, assim como, a explicação respeitante à sua implementação. Relativamente à interface fixa, utilizou-se a linguagem C# da plataforma Micrisoft .NET, que se revelou uma ferramenta poderosíssima para o desenvolvimento de interfaces HMI, visto possuir um enorme conjunto de bibliotecas que permitem: a utilização de botões, caixas de texto, caixas de múltipla escolha, etc.; utilização de portas série; utilização de temporizadores; criação de documentos em diversos formatos (txt, pdf, doc, pp, xl, etc.); manipulação dinâmica de imagens; interligações com bases de dados; envio de mensagens de correio electrónico; encriptação de variáveis; comunicações com Web Site e outras aplicações do Windows (essencialmente se forem desenvolvidas pela plataforma .Net); entre muitas outras. C# revelou-se ainda, uma linguagem muito fácil de utilizar, até mesmo para utilizadores sem qualquer experiência de programação da mesma, como no caso do autor. Porém, a principal limitação desta ferramenta é a sua incapacidade de utilização noutros sistemas operativos, que não os Windows da Microsoft.

No respeitante à utilização da base de dados MySQL, onde são armazenados os dados, pela interface C#, não se verificou qualquer desvantagem, pois esta demonstrou ser uma ferramenta de armazenamento muito eficaz e fácil de utilizar, para conhecedores da linguagem SQL utilizada para manipulação dos dados nestas bases de dados. O único aspecto apontar, não exclusivamente do MySQL mas, de qualquer base de dados remota, é que a velocidade de armazenamento dos dados depende directamente da velocidade da ligação à internet, utilizada no computador onde estiver implementada a interface, portanto se esta for demasiado lenta pode provocar o bloqueio da HMI durante um período de tempo significativo.

Ainda no capítulo 4, é comentada e explicada a utilização do ASP.NET no sistema, que se revelou também uma ferramenta eficaz e fácil de utilizar, para o desenvolvimento de Web sites (interface Web), essencialmente se o seu servidor estiver implementado numa linguagem da plataforma .Net, como é o caso. A grande desvantagem desta linguagem é que, ao contrário da linguagem Javascrip, não é de tempo real, isto é, uma página depois de aberta num *browser* não pode alterada dinamicamente, sendo, para isso, necessário fazer o *refresh* da mesma. No entanto, é possível a utilização de componentes AJAX, como temporizadores, que possibilitam por exemplo, que o *refresh* da página seja realizado autonomamente com um período definido. Apesar de nunca ter sido testado, prevê-se que seja possível que mais que um cliente (browser) possa aceder ao servidor da página Web, sendo criado, de cada vez que um novo computador é conectado, uma nova *thread* para a comunicação entre este e a interface.

Seguidamente, no capítulo 5, foram apresentadas as redes de comunicação utilizadas no sistema protótipo, nomeadamente a rede CAN-Bus e a rede *wireless* 802.15.4. Como foi explicado, a rede CAN-Bus satisfaz todos os requisitos impostos para uma rede de controlo

pretendida nestes sistemas de monitorização, por apresentar velocidades de transmissão; distâncias máximas dos cabos; latência da comunicação; custo; entre outros aspectos, apropriados, visto isto, numa primeira análise parecia ser uma óptima solução para estes sistemas. Todavia, durante a implementação desta rede no sistema, deparam-se alguns problemas para utilização desta rede, sendo todos eles relacionados com o facto de numa trama CAN apenas se poderem utilizar 8 bytes de dados. Esta característica impôs uma complexidade na implementação, não esperada inicialmente, por ser necessário utilizar em muitos casos mais que uma trama CAN para uma troca de dados adequada. A mesma, também limitou no sistema implementado, que o número de sensores que podem ser lidos ao mesmo tempo, seja um valor pequeno (4 no caso), caso contraio, seria necessário utilizar mais tramas CAN o que poderia conduzir a um atraso significativo no tempo de troca das mensagens de leitura, e consequentemente, um bloqueio da rede durante um intervalo de tempo significativo. Concluí-se portanto, que a rede CAN-Bus é, de facto, uma solução possível de se utilizar nestes sistemas, como ficou provado, no entanto, apresenta esta limitação que deve ser considera previamente à sua escolha, por exemplo, para o caso de se querer obter na interface, leituras de mais de 4 sensores ao mesmo tempo e, também, para o caso de se querer ler, de uma só vez (tramas L), vários valores de uma mesmo sensor com bastante frequência. Então, por apresentarem esta limitação, estas redes são normalmente, utilizadas apenas como redes de campo e não como redes de controlo.

Relativamente às redes *wireless*, apresentam-se como uma solução muito barata para sistemas deste género (onde normalmente se abusa da cablagem), não apresentando qualquer limitação no que diz respeito à comunicação em si. Há porém, uma enorme desvantagem no uso de componentes sem fios, que diz respeito à necessidade de utilização de baterias. Deste modo, é necessário projectar muito bem o sistema e nomeadamente, os módulos alimentados pelas baterias, que neste caso são as Unidades de Processo, seus sensores, circuitos de aquisição e os actuadores, pois em alguns casos, pode não ser compensativo o preço das baterias em relação à redução obtida por não se utilizarem cabos. Estas baterias devem conseguir alimentar os dispositivos durante bastante tempo, pois seria muito desvantajoso que em pequenos períodos de tempo fosse necessário substituir ou carregar essas baterias, principalmente para estruturas de grandes dimensões, dado estas poderem-se encontrar em locais de difícil acesso (alturas muito elevadas, por exemplo).

Também no capítulo 5, explicou-se a implementação da Unidade de Comunicação e Controlo Inteligente nos Waspmotes. O processo aqui implementado resulta para este sistema protótipo, porque a uma destas Unidades, estão apenas associadas, no máximo, duas Unidades de Processo. No entanto, para um sistema de maiores dimensões, deve ser considerada a utilização de outros processadores mais robustos, como microprocessadores com sistema operativo, que poderão permitir a realização das tarefas de comunicação e controlo de forma mais rápida e eficiente.

Seguidamente no capítulo 6, foram apresentados alguns dos sensores mais utilizados em sistemas de monitorização de estruturas, tais como, extensómetros, acelerómetros e sensores de temperatura, particularmente os modelos utilizados no protótipo implementado: extensómetros de resistência eléctrica, acelerómetros tipo MEMS e sensores de temperaturas de circuitos integrados. Sendo ainda, referido o circuito de condicionamento de sinal utilizado (adaptado ao ADC dos Waspmotes), bem como, o modelo matemático utilizado para o cálculo das grandezas medidas nas suas unidades representativas. É ainda, apresentado todo o processo implementado na Unidade de Processo, que também se desenvolveu no Waspmote, o

qual se revelou um componente bastante eficiente na realização das tarefas de medição e aquisição dos sensores, que apresenta um ADC incorporado, multiplexado em 7 portas, permitindo que lhe sejam conectados directamente 7 sensores, e ostenta também, 8 saídas digitais às quais podem ser conectados 8 actuadores.

Por fim, no capítulo 7, são apresentados e discutidos alguns resultados do sistema implementado, nomeadamente resultados da velocidade das comunicação, que se revelaram bastante bons para o sistema implementado, isto porque foram praticamente todos menores que 1 segundo, que foi a restrição de tempo real imposta para o sistema.

Foram ainda, realizados ensaios a todos os sensores utilizados, dos quais se pode concluir, pelos resultados obtidos nos mesmos, que tanto os circuitos de condicionamento de sinal desenvolvidos, como os modelos matemáticos encontrados (para os extensómetros e acelerómetros), funcionam correctamente.

Concluí-se portanto de um modo geral, que os objectivos propostos para este trabalho foram alcançados. Estudaram-se e testaram-se variadas tecnologias, de onde se destacam: C#, CAN-Bus e Unidades de sensores *wireless*, que apesar de algumas limitações, particularmente no caso das redes, são uma solução previsivelmente barata e que permitem realizar as tarefas de supervisão e controlo de estruturas. Considera-se portanto, que todos os aspectos abordados ao longo deste documento, se revelam fulcrais e determinantes numa área emergente como é a Civiónica.

8.2 - Desenvolvimentos Futuro

Nesta dissertação foi apresentado um estudo de tecnologias até então, não utilizadas em sistemas de monitorização de estruturas, bem como, o desenvolvimento de um protótipo de um sistema de monitorização completo, utilizando estas mesmas tecnologias. Ficou, no entanto, evidente para o autor que seria bastante interessante a aplicação e teste deste mesmo sistema numa estrutura (ponte real), onde poderiam existir outros problemas, como: o electromagnetismo muito presente nestas estruturas; condições atmosféricas adversas; distância entre elementos reais, por exemplo, a distância de 500m entre gateways que não foi testada, devido ao elevado custo de um cabo CAN para realizar esses testes; entre outros factores.

Relativamente aos sensores, seria interessante utilizar e até desenvolver outros sensores, como: inclinómetros; sensores de corrosão; sensor medidor de flechas (poderia ser desenvolvido um destes sensores por ultra-sons); entre outros.

Por conseguinte, o sistema protótipo implementado, como se verificou (capítulo 7), apresenta resultados bastante interessantes, há no entanto, do ponto de vista do autor, algumas melhorias que poderiam ser realizadas. Uma dessas melhorias seria no protocolo geral, não na sintaxe do protocolo em si mas na forma como é transmitido. Pois, como se sabe, a comunicação é feita ao caracter, isto é, para se enviar o valor “123”, enviam-se os caracteres ‘1’, ‘2’ e ‘3’ (em 3 bytes) sequencialmente, esta transmissão poderia ser mais rapidamente realizada se fosse feita por bytes (como a comunicação CAN), enviando-se num único byte o valor 123.

Poderia também, introduzir-se no sistema outros algoritmos que realizem tarefas de controlo, activando actuadores em função de variáveis monitorizadas, por exemplo, ao verifica-se que a ponte estava em alta ressonância (pelos acelerómetros), poderia alertar-se

os seus utilizadores activando o sinal amarelo do semáforo de forma autónoma, não sendo necessário o operador realizar essa tarefa.

Como foi sendo realçado neste documento, utiliza-se no protótipo implementado, essencialmente o modelo de cooperação Mestre – Escravo, o que impõe algumas limitações temporais, principalmente no que toca à visualização de informações por parte do operador na interface, pois, é necessário que da interface (Mestre) surja, por exemplo, um pedido de leitura dos alarmes, para que os alarmes lhe sejam reportados pelos seus elementos (Escravos). Sugere-se portanto, num futuro desenvolvimento, o teste de um outro modelo de cooperação, por exemplo o Produtor – Consumidor, onde seria necessário um controlo de acesso ao meio mais complexo.

Seriam ainda interessante que as Unidades de Comunicação e Controlo Inteligente e até mesmo as Unidades de Processo comunicassem entre si, por CAN e *wireless*, respectivamente. Desta forma poderiam, reportar entre elas as medições que estão a efectuar, e no caso das Unidades de Processo, conjugadas às informações de outras Unidades, poderiam compor informação mais relevante que uma medição de um sensor isolada. Neste contexto poder-se-ia, ainda, desenvolver por exemplo, um algoritmo de FFT, que calcula-se a frequência de vibração da estrutura pelos valores medidos nos acelerómetros. No caso das Unidades Intermédias, poderiam por exemplo, compreender os alarmes que cada uma contém, enviando à Unidade Central por ordem de prioridades os alarmes que verificaram.

Outro aspecto interessante seria dotar as Unidades Intermédias e Inferiores com memórias de grande capacidade (cartões SD), de modo a poderem guardar durante muito tempo as medições que efectuassem, sem necessidade de as comunicar à Unidade Superior para que sejam armazenadas. Sendo que, esses valores guardados poderiam ser acedidos pela Unidade Superior sempre que o operador desejasse.

Também como se pode verificar, a sincronização dos relógios internos de todos os elementos do sistema, é realizada apenas por uma mensagem de comando enviada pela Unidade de Interface, Comando e Controlo, o que não realizará uma sincronização muito precisa, aliás como se deve imaginar. Então, seria útil a utilização de um algoritmo de sincronização mais completo, como o *Network Time Protocol*.

Relativamente às *gateways* CAN – RS-232, há pelo menos um aspecto que deveria ser melhorado (só não foi devido a condicionantes temporais), que é relativo às comunicações das tramas de leitura de vários valores consecutivos. Pois, se a transmissão não for realizada com sucesso, por exemplo, se uma das tramas não for enviada o programa ficará bloqueado, o que seria uma grande adversidade para um sistema real. Poder-se-ia então, utilizar por exemplo um temporizador, como *timeout*, que libertaria a *gateway* caso não fossem recebidas todas as mensagens num certo intervalo de tempo.

Ainda por condicionantes de tempo, não foi definido nas Unidades de Processo a forma como estas devem sair de *sleep mode*, contudo, os Waspmotes possuem algumas ferramentas que permitem o seu "acordar".

Por fim, há alguns aspectos a melhorar relacionados directamente com os alarmes do sistema. Entre os quais a implementação de um alarme de comunicação, que permitisse ao operador perceber em que parte do sistema pode ter ocorrido um erro, por exemplo, se o erro ocorreu na comunicação *wireless* entre o Mestre 1 e o Escravo 2. Estes erros são no sistema implementado, tratados todos da mesma forma, como um alarme de *timeout* verificado apenas na Unidade superior. Uma outra limitação, verifica-se quando é activado um alarme de threshold, em que a mensagem enviada não contém o identificador do sensor

em que ocorreu esse alarme, infelizmente, também por condicionantes temporais, não foi possível solucionar esta limitação. Finalmente, deveriam junto dos entendidos, Engenheiros Civis, definir-se correctamente os limites de *threshlod* para os alarmes respectivos, bem como, para que situações deveria a interface informar o operador que a ponte se encontra num estado instável, sendo que, no sistema implementado, a ponte é dada como instável sempre que se verifica a ocorrência de uma alarme.

Referências

[1] A. Hejll, "Civil Strutural Health Monitoring - Strategies, Methods and Applications," 2007.

[2] M. P. Groover, *Automation, production systems, and computer-integrated manufacturing* vol. 2nd ed. Englewood Cliffs, NJ: Prentce-Hall, 2001.

[3] J. R. Pimentel, *Communication networks for manufacturing*. Englewood Cliffs, NJ: Prentice-Hall, 1990.

[4] R. Zurawski, *Integration technologies for industrial automated systems*. Boca Raton, FL :: CRC Taylor & Francis, 2007.

[5] P. J. L. M. Portugal, *et al.*, *Avaliação da confiança no funcionamento de redes de campo : contribuição no domínio dos sistemas industriais de controlo*: [s.n.], 2004.

[6] D. Brandl, *Design Patterns for Flexible Manufacturing*. [S. l.]: ISA - The Instrumentation, Systems and Automation Society, 2007.

[7] J. R. P. a. M. Salazar, "Dependability of distributed control system fault," 2002.

[8] S.-T. Levi and A. K. Agrawala, *Real-time system design*. New York: McGraw-Hill Publishing Company, 1990.

[9] E. J. F. Figueiredo, *Monitorização e avaliação do comportamento de obras de arte*. Porto: [s.n.], 2006.

[10] A. M. M. P. Faria, "Avaliação do Desempenho de Sistemas de Monitorização de Estruturas," 2010.

[11] S. Cho, Yun, C.-B., Lynch, J. P., Zimmerman, A. T., Spencer Jr, B. F., & Nagayama, T. , "Smart Wireless Sensor Technology for Structural Health Monitoring of Civil Structures," 2008.

[12] J. W. Dally, *Instrumentation for engineering measurements* vol. 2nd ed 0009. New York [etc]: John Wiley & Sons.

[13] F. J. M. G. d. S. Cavadas, *Monitorização e análise do comportamento de pontes metálicas antigas a Ponte Eiffel*. Porto:: [s. n.], 2008.

[14] A. S. Tanenbaum and M. v. Steen, *Distributed systems principles and paradigms* vol. 2nd ed. Uper Saddle River, NJ: Pearson Prentice Hall, 2007.

[15] Maxim, "Understanding and Using Cyclic Redundancy Checks with Maxim iButton Products," 2001.

[16] A. Rodriguez Penin, *Sistemas Scada* vol. 2.ª ed. Barcelona: Marcombo, Ediciones Técnicas, 2007.

[17] S. Boyer, *Scada supervisory control and data acquisition* vol. 2nd. North Carolina: Instrument Society of America, 1999.

[18] K. Watson, *Beginning visual C# 2005*. Indianapolis, Indiana: Wiley Publishing, 2006.

[19] Digi, *XBee®/XBee-PRO® RF Modules*. Minnesota: Digi International Inc., 2009.

[20] R. Zurawski, *The industrial information technology handbook*. Boca Raton :: CRC Press, 2005.

[21] R. Zurawski, *The industrial communication technology handbook*. Boca Raton: CRC Press, 2005.

[22] J.-P. Thomesse, "Fieldbus Technology in Industrial Automation," 2005.

[23] I. M. G. F. V. Pinheiro, *et al.*, *Sistema de instrumentação distribuída suportado por rede sem fios*: [s. n.], 2008.
[24] R. Bosh, *CAN Specification V2.0*. Stuttgart: Bosh, 1991.
[25] J. M. R. P. d. Sousa, *Aplicação do protocolo CAN a uma rede de comunicações*: FEUP, 2000.
[26] Atmel, *Technical Datasheet of AT90CAN32/64/128*: Atmel Corporation, 2006.
[27] CiA. *CAN in Automation*. Available: http://www.can-cia.org/
[28] G. M. F. Fernandes, *et al.*, *Análise e desenvolvimento de comunicação e computação móvel em sistemas de automação*: FEUP, 2004.
[29] D. Gascón. (2008). *802.15.4 vs ZigBee*. Available: http://www.sensor-networks.org/index.php?page=0823123150
[30] Jennic, *Calculating 802.15.4 Data Rates*, 2006.
[31] A. R. e. Cunha, *et al.*, *On the use of IEEE 802.15.4/ZigBee as federating communication protocols for Wireless Sensor Networks*: [s. n.], 2007.
[32] C. Stiernberg. (2011). *Five Things Every Scientist and Engineer Should Know about Wireless Sensor Measurements*. Available: http://www.laboratoryequipment.com/article-wireless-sensor-measurements-0903.aspx
[33] J. M. a. Y. Lu, "Feature Extraction and Sensor Fusion for Ultrasonic Structural Health Monitoring Under Changing Environmental Conditions," 2009.
[34] C. M. d. S. Félix, *Monitorização e análise do comportamento de obras de arte*. Porto:: FEUP, 2004.
[35] J. Desai. (2007). *Using a strain-gauge transducer in a Wheatstone-bridge configuration*. Available: http://www.eetimes.com/design/analog-design/4009984/Using-a-strain-gauge-transducer-in-a-Wheatstone-bridge-configuration/
[36] F. Cacchione, "Mechanical Characterization and Simulation of fracture processes in polysilicon Micro Electro Mechanical Systems (MEMS)," Doctoral Thesis, Politecnico de Milano, 2007.
[37] Maxim. (2004). *Selecting Temperature Sensors for System Measurement and Protection*. Available: http://www.maxim-ic.com/app-notes/index.mvp/id/3229