

Faculdade de Engenharia da Universidade do Porto



**Migração do Sistema de Controlo Distribuído da Unidade
de Dessulfuração de Gasóleo da Refinaria da GALP de
Matosinhos**

Inês Isabel Pinto Droga

Dissertação realizada no âmbito do
Mestrado em Automação, Instrumentação e Controlo

Orientador: Prof. Dr. Paulo José Lopes Machado Portugal
Co-Orientador: Eng. José Manuel Lluvet Santos

Setembro 2009

Dedicatória

Pela minha Tita.

Em memória de Pedro Romeu Baixinho Rodrigues.

A toda a minha família, em especial ao meu marido e aos meus pais.

Resumo

A refinaria de Matosinhos da Galp Energia S.A. enfrenta o desafio de migrar as configurações do sistema de controlo distribuído *Rosemount System 3*, considerado obsoleto em 2010, para o novo sistema de controlo *DeltaV*.

Neste contexto, esta dissertação teve como objectivo propor novas estratégias para o processo de migração de forma a torná-lo mais automatizado, mais eficiente, mais rápido e sujeito ao menor número de erros possível.

Este processo é de fundamental relevância para as unidades de dessulfuração de gasóleo (*HDS*) e *Claus* - unidade de conversão catalítica de sulfureto de hidrogénio, visto serem alvo de reestruturação, de forma a contemplarem um aumento de capacidade, imposta pela construção de novas unidades de tratamento no complexo da refinaria.

Dada a vastidão do problema em causa, a dissertação foi focada em dois importantes conjuntos de blocos do antigo sistema de controlo: blocos de comunicação com autómatos programáveis (*Programmable Logic Controllers*) e blocos de gestão e operação de válvulas e bombas. A aproximação efectuada a cada um deles foi distinta devido às diferentes características dos dois sistemas de controlo distribuído, nomeadamente no que se refere ao ambiente de desenvolvimento.

Para os blocos de comunicação com autómatos programáveis, foi proposta uma estratégia com base no desenvolvimento de uma aplicação que permite realizar, de uma forma automática, este processo. No caso dos blocos de gestão e operação de válvulas e bombas, foi proposta uma metodologia de migração com base na solução *DeltaV*.

A execução destes dois objectivos foi cumprida e, alguns dos blocos provenientes da conversão automática, foram testados em ambiente industrial e os módulos de gestão de válvulas e bombas já se encontram perfeitamente operacionais.

Abstract

The oil refinery of Galp Energia S. A. in Matosinhos faces the challenge of bringing forward the migration of the configurations of the distributed control system, *Rosemount System 3*, taken as obsolete in 2010, into the new control system *DeltaV*.

In this context, the present dissertation main goal was to propose new strategies that enable the migration process to be more automatized, more efficient, faster, and less prone to mistakes.

The process is of fundamental relevance for the gas hydrodesulfurization unit (*HDS*) and *Claus* unit (catalytic conversion of hydrogen sulphur), as there are restructuration targets, in order for them to cope with the capacity increment imposed by the construction of new units in the refinery complex.

Given the vastness of this matter, this dissertation focused in two important groups of blocks: communication blocks with programmable logic controllers and blocks for management and operation of valves and pumps. The approach made to each of them was distinct due to the different characteristics of both distributed control systems, namely in what respects to their development environment.

In the case of the blocks for communication with programmable logic controllers, it was proposed a strategy based on the development of an application to automatically accomplish the process. In the case of the blocks for management and operation of valves and pumps, a methodology of migration based on the *DeltaV* solution was proposed.

The execution of those two objectives was fulfilled and some units originated from automatic conversion were tested in industrial environment, and the modules for management of valves and pumps are already fully operative.

Agradecimentos

Ao meu marido.

Aos meus pais.

Ao Eng. Manuel Barreira, Eng. Manuel Carvalho e Eng. Luís Colmonero pela oportunidade e confiança que depositaram em mim.

Ao Prof. Dr. Paulo Portugal pela paciência e dedicação.

Ao Eng. Lluvet Santos pela disponibilidade.

Índice

DEDICATÓRIA.....	III
RESUMO	V
ABSTRACT.....	VII
AGRADECIMENTOS.....	VII
ÍNDICE	XI
LISTA DE FIGURAS	XV
LISTA DE TABELAS.....	XIX
LISTA DE ACRÓNIMOS.....	XXI
CAPÍTULO 1.....	1
INTRODUÇÃO E OBJECTIVOS	1
1.1. <i>Introdução</i>	1
1.2. <i>Organização do Documento</i>	2
CAPÍTULO 2.....	5
GALP ENERGIA.....	5
2.1. <i>O Grupo Galp Energia</i>	5
2.2. <i>Refinaria de Matosinhos</i>	6
2.3. <i>Fábrica de Combustíveis</i>	8
2.3.1. Unidade de Dessulfuração de Gasóleo - UN 3700	10
2.3.2. Unidade de Conversão Catalítica de Sulfureto de Hidrogénio - UN 3800.....	11
CAPÍTULO 3.....	13
SISTEMAS DE CONTROLO DISTRIBUÍDO.....	13
3.1. <i>Introdução</i>	13
3.2. <i>Evolução Histórica</i>	15
3.3. <i>Arquitectura</i>	19
3.4. <i>Aplicações Típicas de um DCS</i>	22
CAPÍTULO 4.....	23
SISTEMAS DE CONTROLO DISTRIBUÍDO NA REFINARIA DE MATOSINHOS	23
4.1. <i>Rosemount System 3 - RS3</i>	23
4.1.1. <i>PeerWay</i>	24

4.1.2. ControlFiles	26
4.1.3. Cartas de E/S	28
4.1.4. Consolas de Operação e Configuração	29
4.1.5. Interface com Equipamentos Externos	31
4.1.6. Estrutura e Funcionalidades do <i>Software</i>	31
4.1.7. Endereçamento.....	37
4.1.8. RS3 na refinaria de Matosinhos	39
4.2. DeltaV.....	40
4.2.1. Barramento de comunicação.....	41
4.2.2. Controladores.....	42
4.2.3. Cartas de E/S	44
4.2.4. Estações de trabalho	46
4.2.5. Estrutura e Funcionalidades do <i>Software</i>	47
4.2.6. Endereçamento.....	50
4.2.7. DeltaV na refinaria de Matosinhos	50
CAPÍTULO 5.....	53
DISCUSSÃO DO PROBLEMA	53
5.1. Ambiente de Desenvolvimento do RS3	53
5.2. Ambiente de Desenvolvimento do DeltaV.....	55
5.3. Processo de Migração do RS3 para o DeltaV	58
5.3.1. Processo de Migração de Configurações de PLC.....	63
5.3.2. Processo de Migração de Configuração de Válvulas e Bombas	69
CAPÍTULO 6.....	75
ESPECIFICAÇÃO DA SOLUÇÃO PROPOSTA	75
6.1. Especificação	75
6.1.1. Requisitos	76
6.1.1.1. Análise de Requisitos - Migração de Configurações de PLC	77
6.1.1.2. Análise de Requisitos - Configuração de Válvulas e Bombas	85
6.1.2. Solução Proposta	86
6.1.2.1. Migração da Configuração de PLC.....	86
6.1.2.2. Modelação da Aplicação em UML	90
6.1.2.3. Migração da Configuração de Válvulas e Bombas.....	95
CAPÍTULO 7.....	99
IMPLEMENTAÇÃO E TESTES	99
7.1. Desenvolvimento da Aplicação.....	99
7.2. Interface da Aplicação com o Utilizador	100
7.3. Implementação dos Módulos para Válvulas e Bombas	106
7.3.1. Desenvolvimento dos Módulos e Faceplates	106
7.3. Testes da Aplicação.....	113
7.3.1. Testes Preliminares.....	113
7.3.2. Testes de Validação	114
7.4. Testes dos Módulos para Válvulas e Bombas	115

7.4.1. Testes Preliminares.....	115
7.4.2. Testes de Validação	115
CAPÍTULO 8.....	117
CONCLUSÕES	117
ANEXO 1.....	I
BLOCOS FUNÇÃO DO SISTEMA <i>RS3</i>	I
ANEXO 2.....	I
BLOCOS FUNÇÃO DO SISTEMA <i>DELTA V</i>	I
ANEXO 3.....	I
APLICAÇÕES DO SISTEMA <i>DELTA V</i>	I
ANEXO 4.....	I
BLOCO <i>DMC</i> DO <i>RS3</i>	I
ANEXO 5.....	I
FICHEIRO DE ESPECIFICAÇÃO DE FORMATO	I
ANEXO 6.....	I
PARÂMETROS DA APLICAÇÃO.....	I
ANEXO 7.....	I
PARÂMETROS DA APLICAÇÃO.....	I
ANEXO 8.....	I
FASE I - TESTES EFECTUADOS	I
ANEXO 9.....	I
FASE II - TESTES EFECTUADOS	I

Lista de Figuras

FIGURA 2.1 - FOTOGRAFIA AÉREA DA REFINARIA DE MATOSINHOS [1].	7
FIGURA 2.2 - LOCALIZAÇÃO DA FÁBRICA DE COMBUSTÍVEIS NO COMPLEXO DA REFINARIA DE MATOSINHOS [1].	9
FIGURA 2.3 - DIAGRAMA SIMPLIFICADO DAS UNIDADES PROCESSUAIS DA FÁBRICA DE COMBUSTÍVEIS [1].	9
FIGURA 2.4 - ESQUEMA SIMPLIFICADO DA UN3700.	11
FIGURA 2.5 - ESQUEMA SIMPLIFICADO DA UN3800.	12
FIGURA 3.1 - SISTEMA DE CONTROLO CENTRALIZADO [3].	14
FIGURA 3.2 - SISTEMA DE CONTROLO DESCENTRALIZADO [3].	14
FIGURA 3.3 - SISTEMA DE CONTROLO DISTRIBUÍDO.	15
FIGURA 3.4 - IBM 1800: AQUISIÇÃO DE DADOS E SISTEMA DE CONTROLO CENTRALIZADO [7].	16
FIGURA 3.5 - CONTROLADORES DO DCS TDC 2000 DA HONEYWELL [8].	17
FIGURA 3.6 - HMI DO DCS CENTUM DA YOKOGAWA [9].	17
FIGURA 3.7 - EVOLUÇÃO DOS DCS DA YOKOGAWA DESDE O LANÇAMENTO DO CENTUM EM 1975 ATÉ AO CENTUM VP EM 2008 [9].	18
FIGURA 3.8 - ARQUITECTURA FÍSICA OU DE HARDWARE DO SISTEMA EXPERION PKS DA HONEYWELL [11].	20
FIGURA 3.9 - FIABILIDADE VS. FUNCIONALIDADE DE PC, PLC E PAC [13].	22
FIGURA 4.1 - ARQUITECTURA DO SISTEMA RS3 [14].	24
FIGURA 4.2 - COMUNICAÇÃO ENTRE PEERWAYS VIA HIA [14].	25
FIGURA 4.3 - PEERWAY TAPS REDUNDANTES [15].	25
FIGURA 4.4 - NÓS LIGADOS ÀS PEERWAY TAP REDUNDANTES E LIGAÇÃO AO BARRAMENTO PEERWAY [14].	26
FIGURA 4.5 - CARTAS CONSTITUINTES DA CONTROLFILE [14].	26
FIGURA 4.6 - INTERACÇÃO FUNCIONAL ENTRE AS CARTAS DE SUPORTE DAS MPC [15].	28
FIGURA 4.7 - LIGAÇÃO DAS CARTAS E/S À CONTROLFILE E AO PEERWAY [14].	29
FIGURA 4.8 - FACEPLATES DISCRETOS E CONTÍNUOS DO RS3 [14].	30
FIGURA 4.9 - TENDÊNCIA DO RS3 [14].	30
FIGURA 4.10 - LIGAÇÃO DE RNI À REDE ETHERNET E AO PEERWAY [15].	31
FIGURA 4.11 - BLOCO DE ENTRADA ANALÓGICA SMART.	32
FIGURA 4.12 - BLOCO DE SAÍDA ANALÓGICA.	33
FIGURA 4.13 - LOOP DE CONTROLO CONTÍNUO TÍPICO DO RS3.	34
FIGURA 4.14 - LOOP DE CONTROLO DISCRETO TÍPICO DO RS3.	34
FIGURA 4.15 - DIAGRAMA FUNCIONAL DOS BLOCOS DE CONTROLO [14].	35
FIGURA 4.16 - PASSO LÓGICO A DE UM BLOCO DE CONTROLO - SET E CLEAR E ACÇÕES EM ON E OFF COM UM COMENTÁRIO ADICIONADO NO RISE.	36
FIGURA 4.17 - PASSO LÓGICO A DE UM BLOCO DE CONTROLO - CÁLCULOS.	37
FIGURA 4.18 - ENDEREÇO TÍPICO DE UM BLOCO DE ENTRADA OU SAÍDA.	38
FIGURA 4.19 - ENDEREÇO TÍPICO DE UM BLOCO DE CONTROLO.	38

FIGURA 4.20 - ARQUITECTURA DO <i>RS3</i> NA REFINARIA DE MATOSINHOS.	39
FIGURA 4.21 - NÓS DO <i>PEERWAY 2</i>	40
FIGURA 4.22 - ARQUITECTURA DE <i>HARDWARE</i> BÁSICA DO <i>DELTA V</i> [16].	41
FIGURA 4.23 - REDE DE COMUNICAÇÃO COM OITO NÓS: QUATRO CONTROLADORES E QUATRO ESTAÇÕES DE TRABALHO [16].	42
FIGURA 4.24 - CONTROLADORES E FONTES DE ALIMENTAÇÃO DO SISTEMA REDUNDANTES MONTADOS EM CALHA <i>DIN</i> [16].	43
FIGURA 4.25 - ALIMENTAÇÃO DE <i>CARRIERS</i> A PARTIR DAS FONTES DE ALIMENTAÇÃO DO SISTEMA E A PARTIR DA FONTE DE ALIMENTAÇÃO EXTERNA [16].	44
FIGURA 4.26 - <i>TERMINAL BLOCKS</i> E CARTAS <i>E/S</i> MONTADAS NUM <i>CARRIER</i> DE OITO POSIÇÕES [16].	45
FIGURA 4.27 - ALIMENTAÇÃO DOS DISPOSITIVOS DE CAMPO A PARTIR DA FONTE DE ALIMENTAÇÃO EXTERNA DE 24 V <i>DC</i> [16].	45
FIGURA 4.28 - COMPONENTES BÁSICOS DE UM MÓDULO DE <i>DELTA V</i> [16].	47
FIGURA 4.29 - HIERARQUIA DO <i>DELTA V</i>	48
FIGURA 4.30 - BLOCO DE FUNÇÃO DE SAÍDA DIGITAL COM TRÊS PARÂMETROS VISÍVEIS [16].	49
FIGURA 4.31 - ESTRATÉGIA DE CONTROLO SIMPLES ONDE UMA ENTRADA ANALÓGICA ACTUA EM DUAS SAÍDAS ANALÓGICAS.	49
FIGURA 4.32 - ARQUITECTURA DA REDE PRIMÁRIA DO <i>DELTA V</i> DA REFINARIA.	51
FIGURA 5.1 - CONFIGURAÇÃO DE UMA ENTRADA ANALÓGICA <i>AIB</i> NO <i>RS3</i>	54
FIGURA 5.2 - CONFIGURAÇÃO DE UMA ENTRADA ANALÓGICA <i>AIB</i> NO <i>RS3</i> E VALORES COLOCADOS POR DEFEITO LOGO APÓS A CRIAÇÃO DO BLOCO.	54
FIGURA 5.3 - CONFIGURAÇÃO DE UM CANAL PARA ENTRADA ANALÓGICA NA CARTA 5 DO CONTROLADOR <i>CTLR-FUT-2</i> DO <i>DELTA V</i>	56
FIGURA 5.4 - CRIAÇÃO DE UM NOVO MÓDULO NO <i>DELTA V</i>	57
FIGURA 5.5 - <i>CONTROL STUDIO</i> DO <i>DELTA V</i> : CONFIGURAÇÃO DE UMA ENTRADA ANALÓGICA (<i>AI1</i>) A PARTIR DE UM TRANSMISSOR INTELIGENTE (<i>HART_FIELD_VAL</i>).	57
FIGURA 5.6 - BIBLIOTECA COM OS MÓDULOS TÍPICOS CRIADOS NO <i>DELTA V</i> PARA A REFINARIA.	58
FIGURA 5.7 - FICHEIROS DE ESPECIFICAÇÃO DE FORMATO EXISTENTES NO <i>DELTA V</i>	59
FIGURA 5.8 - EXPORTAÇÃO DE MÓDULOS SEGUNDO O FORMATO ESPECIFICADO NA FIG. 5.2.	60
FIGURA 5.9 - CONTEÚDO DE UM FICHEIRO DE ESPECIFICAÇÃO DE FORMATO.	60
FIGURA 5.10 - IMPORTAÇÃO USANDO O FORMATO DE ESPECIFICAÇÃO <i>STANDARD</i> DO <i>DELTA V</i>	61
FIGURA 5.11 - IMPORTAÇÃO USANDO O FORMATO DE ESPECIFICAÇÃO <i>STANDARD</i> DO <i>DELTA V</i>	62
FIGURA 5.12 - BLOCO DE ENTRADA DE <i>FLAGS</i> DE COMUNICAÇÃO <i>MODBUS</i>	63
FIGURA 5.13 - BLOCO DE ENTRADA DE <i>FLAGS</i> DE COMUNICAÇÃO <i>MODBUS</i> : SELECIONADO O <i>BIT</i> A COM O RESPECTIVO ALARME E DESCRITIVO ASSOCIADO.	64
FIGURA 5.14 - BLOCO DE ENTRADA DE <i>U INT</i> DE COMUNICAÇÃO <i>MODBUS</i>	64
FIGURA 5.15 - DIAGRAMA DE PASSOS SEQUENCIAL DA PASSAGEM DE <i>PLC</i> DE <i>RS3</i> PARA <i>DELTA V</i>	66
FIGURA 5.16 - CONFIGURAÇÃO DOS REGISTOS AGRUPADOS EM <i>DATASETS</i> NO <i>DELTA V</i>	67
FIGURA 5.17 - MÓDULO BASE DE LEITURA DO <i>DELTA V</i> QUE LÊ PARÂMETROS DE UM <i>DATASET</i>	67
FIGURA 5.18 - MÓDULO INDIVIDUAL (APENAS UM PARÂMETRO) DE <i>DELTA V</i> COM ALARMES PARA SINAL DIGITAL.	68
FIGURA 5.19 - MÉTODO PARA DESPISTAR E SOLUCIONAR OS ERROS NA MIGRAÇÃO MANUAL DE <i>PLC</i>	69
FIGURA 5.20 - BLOCO <i>DMC</i> DO <i>RS3</i> USADO PARA VÁLVULAS E BOMBAS.	70
FIGURA 5.21 - BLOCO <i>DMC</i> COM OS PARÂMETROS DE CONFIGURAÇÃO COLOCADOS POR DEFEITO.	71
FIGURA 5.22 - MÓDULO COM O BLOCO DE FUNÇÃO <i>DC</i> DO <i>DELTA V</i> USADO PARA VÁLVULAS E BOMBAS.	72
FIGURA 5.23 - SINÓPTICO DE OPERAÇÃO USANDO A APLICAÇÃO <i>DELTA V OPERATE</i>	73
FIGURA 5.24 - <i>FACEPLATE</i> PARA O BLOCO <i>DC</i> E RESPECTIVO <i>FACEPLATE</i> DE DETALHE.	74
FIGURA 6.1 - ENGENHARIA DE PROCESSO DE UMA APLICAÇÃO [21].	76
FIGURA 6.2 - DESENVOLVIMENTO EVOLUTIVO DE UMA APLICAÇÃO [21].	76
FIGURA 6.3 - PROCESSO DE CORRECÇÃO DE ERROS [21].	77

FIGURA 6.4 - PARÂMETROS NECESSÁRIOS DOS BLOCOS <i>PLCB</i> PARA A FASE I - LEITURA TIPO <i>FLAGS</i> .	78
FIGURA 6.5 - PARÂMETROS NECESSÁRIOS DOS BLOCOS <i>PLCB</i> PARA A FASE I - LEITURA TIPO <i>U INT</i> .	78
FIGURA 6.6 - PARÂMETROS NECESSÁRIOS DOS BLOCOS <i>PLCB</i> PARA A FASE I - ESCRITA TIPO <i>FLAGS</i> .	79
FIGURA 6.7 - PARÂMETROS NECESSÁRIOS DOS BLOCOS <i>PLCB</i> PARA A FASE I - ESCRITA TIPO <i>U INT</i> .	79
FIGURA 6.8 - DIVISÃO DOS BLOCOS <i>PLCB</i> PARA A FASE I.	80
FIGURA 6.9 - <i>LANDING MODULE</i> TIPO <i>READ_1</i> - NÚMERO MÁXIMO DE PARÂMETROS É 10.	82
FIGURA 6.10 - <i>LANDING MODULE</i> TIPO <i>READ</i> - NÚMERO MÁXIMO DE PARÂMETROS É 50.	82
FIGURA 6.11 - <i>LANDING MODULE</i> TIPO <i>WRITE_1</i> - NÚMERO MÁXIMO DE PARÂMETROS É 10.	83
FIGURA 6.12 - <i>LANDING MODULE</i> TIPO <i>WRITE</i> - NÚMERO MÁXIMO DE PARÂMETROS É 50.	83
FIGURA 6.13 - MÓDULO INDIVIDUAL TIPO <i>DI</i> PARA LEITURAS DISCRETAS.	84
FIGURA 6.14 - MÓDULO INDIVIDUAL TIPO <i>AI</i> PARA LEITURAS ANALÓGICAS.	85
FIGURA 6.15 - PRINCÍPIO DE FUNCIONAMENTO DA APLICAÇÃO DE MIGRAÇÃO DE BLOCOS <i>PLCB</i> .	88
FIGURA 6.16 - DIVISÃO DA APLICAÇÃO EM DUAS FASES COMPLEMENTARES.	90
FIGURA 6.17 - DIAGRAMA DE <i>USE CASES</i> DA APLICAÇÃO.	91
FIGURA 6.18 - DIAGRAMA DE CLASSES DA APLICAÇÃO.	92
FIGURA 6.19 - DIAGRAMA DE CLASSES DA APLICAÇÃO (CONTINUAÇÃO).	93
FIGURA 6.20 - MÁQUINA DE ESTADOS DA APLICAÇÃO.	94
FIGURA 6.21 - PRINCÍPIO DE FUNCIONAMENTO DO MÓDULO PARA GESTÃO DE VÁLVULAS E BOMBAS.	96
FIGURA 6.22 - PAPEL DO OPERADOR NA GESTÃO DE VÁLVULAS E BOMBAS A PARTIR DO <i>FACEPLATE</i> .	97
FIGURA 7.1 - PRIMEIRA JANELA DA APLICAÇÃO PARA SELECÇÃO DA BASE DE DADOS DO <i>CS FOR RS3</i> .	101
FIGURA 7.2 - SEGUNDA JANELA PARA SELECÇÃO DO PROJECTO E CONFIGURAÇÃO DE PARÂMETROS ADICIONAIS PARA CRIAÇÃO DE <i>DATASETS</i> E MÓDULOS.	102
FIGURA 7.3 - SEGUNDA JANELA PARA SELECÇÃO DO PROJECTO E CONFIGURAÇÃO DE PARÂMETROS ADICIONAIS PARA CRIAÇÃO DE <i>DATASETS</i> E MÓDULOS.	103
FIGURA 7.3 - JANELA INFORMATIVA DE FINALIZAÇÃO DA CRIAÇÃO DE <i>DATASETS</i> .	103
FIGURA 7.4 - JANELA PARA SELECÇÃO DE REGISTOS CONSECUTIVOS OU NÃO QUANDO EXISTE APENAS UM BLOCO DE LEITURA OU ESCRITA.	104
FIGURA 7.5 - PORMENOR DO PARÂMETRO INFORMATIVO DA CRIAÇÃO DE <i>DATASET</i> .	104
FIGURA 7.6 - PORMENOR DO PARÂMETRO INFORMATIVO DA CRIAÇÃO DE MÓDULOS ONDE CONSTA O NÚMERO DO REGISTO, O NÚMERO DE REGISTOS A PROCESSAR E O REGISTO ACTUAL EM PROCESSAMENTO.	105
FIGURA 7.7 - CONTEÚDO PARCIAL DA PASTA <i>DOCS</i> ONDE SÃO COLOCADOS TODOS OS FICHEIROS <i>.TXT</i> E <i>.FHX</i> .	106
FIGURA 7.8 - MÓDULO PARA OPERAÇÃO DE VÁLVULAS E BOMBAS NO <i>DELTA V</i> .	108
FIGURA 7.9 - <i>FACEPLATE</i> PARA OPERAÇÃO DE BOMBA NO <i>DELTA V</i> COM EXPLICAÇÃO DAS FUNCIONALIDADES.	109
FIGURA 7.10 - <i>FACEPLATE</i> DE DETALHE PARA OPERAÇÃO DE BOMBA NO <i>DELTA V</i> .	110
FIGURA 7.11 - CAIXA DE DIÁLOGO REFERENTE À ANIMAÇÃO DE PARÂMETROS E OBJECTOS NO <i>DELTA V OPERATE</i> .	111
FIGURA 7.12 - PORMENOR DO BLOCO <i>DC</i> NO MÓDULO DE <i>DELTA V</i> .	111
FIGURA 7.13 - OPÇÕES DO <i>DC</i> NO <i>DELTA V</i> .	112
FIGURA 7.14 - RESULTADO DA IMPORTAÇÃO DO FICHEIRO <i>DATASETS</i> PARA O <i>DELTA V</i> .	114
FIGURA 7.15 - RESULTADO DA IMPORTAÇÃO DO FICHEIRO <i>DATASETS</i> PARA O <i>DELTA V</i> .	114

Lista de Tabelas

TABELA 2.1 - CAPACIDADES DE FABRICO DA REFINARIA DE MATOSINHOS [1].	8
TABELA 5.1 - LINKS PARA A CONFIGURAÇÃO DE UM BLOCO <i>DMC</i> [20].	71
TABELA 6.1 - ENDEREÇOS <i>MODBUS</i> E RESPECTIVA DESCRIÇÃO NO <i>DELTA V</i> .	80
TABELA 4.1 - BLOCOS DE ENTRADA E SAÍDA DO SISTEMA <i>RS3</i> .	I
TABELA 4.2 - BLOCOS DE CONTROLO - FUNÇÕES CONTÍNUAS - DO SISTEMA <i>RS3</i> .	III
TABELA 4.3 - BLOCOS DE CONTROLO - FUNÇÕES DISCRETAS - DO SISTEMA <i>RS3</i> .	III
TABELA 4.4 - BIBLIOTECA DE BLOCOS DE FUNÇÃO DO SISTEMA <i>DELTA V</i> (VERSÃO 9.3.1).	I
TABELA 4.5 - FERRAMENTAS DE ENGENHARIA DO SISTEMA <i>DELTA V</i> (VERSÃO 9.3.1).	I
TABELA 4.6 - APLICAÇÕES DE OPERAÇÃO SISTEMA <i>DELTA V</i> (VERSÃO 9.3.1).	II
TABELA 4.7 - APLICAÇÕES DE CONTROLO AVANÇADO DO SISTEMA <i>DELTA V</i> (VERSÃO 9.3.1).	III
TABELA 5.2 - CONFIGURAÇÃO DE OPÇÕES DO BLOCO <i>DMC</i> DO <i>RS3</i> .	I
TABELA 5.3 - LINKS DISCRETOS E PASSOS LÓGICOS DO BLOCO <i>DMC</i> DO <i>RS3</i> .	III
TABELA 5.4 - LINKS CONTÍNUOS DO BLOCO <i>DMC</i> DO <i>RS3</i> .	IV
TABELA 6.2 - PARÂMETROS DO FICHEIRO TEXTO E SUA PROVENIÊNCIA - CRIAÇÃO DE <i>DATASETS</i> .	I
TABELA 6.3 - ENDEREÇOS <i>MODBUS</i> E RESPECTIVA DESCRIÇÃO NO <i>DELTA V</i> .	II
TABELA 6.3 - PARÂMETROS DOS FICHEIROS <i>.FHX</i> E SUA PROVENIÊNCIA.	I
TABELA 7.1 - FASE I: TESTES EFECTUADAS PARA A CRIAÇÃO DE <i>DATASETS</i> .	II
TABELA 7.2 - FASE I: TESTES EFECTUADAS PARA A CRIAÇÃO DE <i>DATASETS</i> (CONTINUAÇÃO).	III
TABELA 7.3 - FASE I: TESTES EFECTUADAS PARA A CRIAÇÃO DE FICHEIROS <i>FHX</i> .	II

Lista de Acrónimos

<i>AIB</i>	<i>Analog Input Block</i>
<i>AOB</i>	<i>Analog Output Block</i>
<i>BFI</i>	<i>Boolean Fan In</i>
<i>BFO</i>	<i>Boolean Fan Out</i>
<i>CB</i>	<i>RS3 Control Block</i>
<i>CIB</i>	<i>Contact Input Block</i>
<i>CP</i>	<i>Controller Processor RS3 Card</i>
<i>CS</i>	<i>Control Studio</i>
<i>CSC</i>	<i>Control Studio File</i>
<i>CTLR</i>	<i>Controller</i>
<i>DC</i>	<i>Direct Current</i>
<i>DC</i>	<i>Device Control</i>
<i>DCS</i>	<i>Distributed Control Systems</i>
<i>DDC</i>	<i>Direct Digital Control</i>
<i>DIB</i>	<i>Discrete Input Block</i>
<i>DISC</i>	<i>Discrete Control Block</i>
<i>DMC</i>	<i>Discrete Motor Controller</i>
<i>DOB</i>	<i>Discrete Output Block</i>
<i>E/S</i>	<i>Entrada/Saída</i>
<i>ETAR</i>	<i>Estação de Tratamento de Águas Residuais</i>
<i>FAR</i>	<i>Fábrica de Aromáticos</i>
<i>FCO</i>	<i>Fábrica de Combustíveis</i>
<i>FHX</i>	<i>DeltaV Configuration Files</i>
<i>FIC</i>	<i>Field Interface Card</i>
<i>FIM</i>	<i>Field Interface Module</i>
<i>FMT</i>	<i>Format Specification File</i>

<i>FOB</i>	<i>Fábrica de Óleos Base</i>
<i>FUT</i>	<i>Fábrica de Utilidades</i>
<i>HDS</i>	<i>Hydrodesulfurization unit</i>
<i>HIA</i>	<i>Highway Interface Adaptor</i>
<i>HMI</i>	<i>Human Machine Interface</i>
<i>LUB</i>	<i>Fábrica de Lubrificantes</i>
<i>MAN</i>	<i>Manual Control Block</i>
<i>MATH</i>	<i>Math Control Block</i>
<i>MOV</i>	<i>Movimentação de Produtos</i>
<i>MPC</i>	<i>Multi Propose Controller</i>
<i>MTCC</i>	<i>Mutitube Command Console</i>
<i>OLE</i>	<i>Object Linking and Embedding</i>
<i>OPC</i>	<i>OLE for Process Control</i>
<i>PC</i>	<i>Personal Computer</i>
<i>PLC</i>	<i>Programmable Logic Controller</i>
<i>PLCB</i>	<i>Programmable Logic Control Block</i>
<i>PW</i>	<i>RS3 PeerWay</i>
<i>RNI</i>	<i>Rosemount Network Interface</i>
<i>RS3</i>	<i>Rosemount System 3</i>
<i>SCLR</i>	<i>Scaler</i>
<i>SIB</i>	<i>Smart Input Block</i>
<i>TCP/IP</i>	<i>Transmission Control Protocol/ Internet Protocol</i>
<i>TP</i>	<i>Timed Pulse</i>
<i>TPL</i>	<i>Terminal do Porto de Leixões</i>
<i>UML</i>	<i>Unified Modelling Language</i>

Capítulo 1

Introdução e Objectivos

Neste capítulo são apresentadas as considerações básicas relativas à elaboração desta dissertação, os objectivos que se pretendem atingir e é referida a organização estrutural deste documento.

1.1. Introdução

O complexo da refinaria de Matosinhos, pertencente à Galp Energia S.A., atravessa uma fase de reestruturação devido à construção de novas unidades processuais bem como ao *revamping*⁽¹⁾ das unidades existentes.

Para que o complexo actual acompanhe esta reestruturação é necessário proceder a inúmeras alterações na base instalada, nomeadamente ao nível dos sistemas de controlo distribuído. Existem actualmente dois sistemas deste tipo implementados na refinaria de Matosinhos com filosofias distintas ao nível de arquitectura de *hardware* e de *software*: o *Rosemount System 3 (RS3)* e o *DeltaV*, ambos da *Emerson Process Management*. A grande maioria dos equipamentos e instrumentos são controlados e monitorizados utilizando um destes sistemas de controlo, que possuem uma interface gráfica de operação que acede a ambas as bases de dados e, a partir da qual, as unidades processuais conduzidas por operadores experientes.

Sendo que a primeira instalação do *RS3* em 1991 na fábrica de combustíveis, seguida pelas restantes fábricas em anos subsequentes, o sistema será considerado obsoleto em 2010 pelo que é necessário efectuar uma migração de todo o *hardware* e *software* para um novo sistema.

Neste sentido, a refinaria iniciou, em 2005, a migração do *RS3* para o novo sistema *DeltaV*. Gradualmente todos os sinais físicos e de comunicação com outros sistemas como é o caso dos *PLC (Programmable Logic Controller)* serão realocados fisicamente. A reestruturação acima referida veio antecipar algumas das alterações previstas, ou seja, acelerou o processo de

⁽¹⁾ - *revamping* - reorganização estrutural de uma ou mais unidades processuais para contemplar um aumento de capacidade produtiva ou para ir ao encontro de alterações impostas nas matérias-primas

migração para unidades alvo de *revamping*, nomeadamente da unidade de dessulfuração de gasóleo (*HDS*) e a *Claus* - unidade de conversão catalítica de sulfureto de hidrogénio.

Assim sendo, todos os novos projectos e ampliações que necessitem de ligação de sinais físicos ao sistema de controlo distribuído serão realizados em *DeltaV*. A comunicação com outros sistemas que é usualmente efectuada recorrendo ao protocolo *MODBUS*, será também realizada utilizando a nova interface disponibilizada pelo *DeltaV* para a comunicação série.

Actualmente, a migração de todos os módulos é efectuada manualmente, transpondo a configuração do sistema antigo para o novo sistema, com os vários problemas que este processo pode trazer, nomeadamente no que diz respeito a erros humanos: esquecimento de configuração de alarmes, troca de escalas, unidades ou número de registo *MODBUS*. Em alguns casos estes erros são graves visto que, processualmente, o operador não terá a noção real de como se encontra o equipamento, processo ou unidade.

Para além dos diversos erros que podem ser cometidos pelo configurador, existe a componente tempo que é muito significativa. A configuração manual é demasiado morosa, pois requer muito tempo para recolha de informação e reconfiguração no novo sistema: por exemplo, o tempo necessário para realizar a migração das configurações de comunicação de 300 registo *MODBUS* é cerca de 35 horas.

De forma a colmatar estes problemas, esta dissertação tem como objectivos:

1. Desenvolver uma aplicação informática para migração, o mais automatizada possível, dos blocos de comunicação que utilizam o protocolo *MODBUS* desde o *RS3* até ao *DeltaV*. Este tipo de blocos é de vital importância visto que, comunicam com *PLC* de seguranças processuais, gestão de motores ou *Fire&Gas*, sistemas estes considerados críticos.
2. Desenvolver uma nova estratégia de controlo para blocos do *RS3* relacionados com a operação de bombas e válvulas (devido à génese dos dois sistemas não é possível a implementação de uma ferramenta automática para todos os blocos) e, respectiva migração para o *DeltaV*.

Neste último caso, serão necessários construir novas interfaces de operação destes equipamentos, ou seja, serão necessários construir novos *faceplates* que permitam a sua correcta operação a partir da sala de controlo. Habitualmente, para cada tipo de módulo, existe um *faceplate* de operação e outro *faceplate* que contém detalhes e informações adicionais sobre o equipamento. Estes dois tipos de *faceplates* foram também desenvolvidos no âmbito desta dissertação.

De salientar que a execução destes objectivos serão úteis na migração de sinais de todas as fábricas do complexo (não só das unidades *HDS* e *Claus*), permitindo assim, a redução do tempo de execução da programação em *DeltaV*.

1.2. Organização do Documento

A dissertação está organizada em oito capítulos, sendo o primeiro a introdução. No segundo capítulo é feito o enquadramento do trabalho realizado, focando as principais actividades da Galp Energia S.A., bem como uma breve descrição da refinaria de Matosinhos e das unidades processuais alvo deste trabalho.

No terceiro capítulo é feita uma descrição, ainda que breve, da evolução dos sistemas de controlo distribuído: arquitecturas físicas e de *software*, interfaces, aplicações típicas destes sistemas.

No quarto capítulo descrevem-se, com detalhe, os dois sistemas de controlo distribuído, *RS3* e *DeltaV*, ao nível da arquitectura física e da estrutura e funcionalidades do *software*.

O ambiente de desenvolvimento de ambos os sistemas é discutido no capítulo cinco, onde se pretende detalhar os métodos utilizados actualmente nas migrações em curso na refinaria e identificar quais os problemas que se colocam nos processos de migração.

A especificação da solução da qual fazem parte a análise de requisitos e o princípio de funcionamento da aplicação estão no capítulo seis, seguido dos detalhes da sua implementação e testes de validação no capítulo sete.

O capítulo oito apresenta as conclusões retiradas após o cumprimento dos objectivos.

Capítulo 2

Galp Energia

Este capítulo pretende apresentar o grupo Galp Energia e as suas actividades no sector industrial e económico do país, passando por uma breve descrição das principais empresas que integram o grupo.

Será focada a única empresa refinadora do país, a Petrogal, a refinaria de Matosinhos, que a integra e serão descritas, de forma breve, as duas unidades processuais alvo desta dissertação, de forma a enquadrá-la.

2.1. O Grupo Galp Energia

A Galp Energia, SGPS, SA foi constituída em 1999, sob a denominação GALP - Petróleos e Gás de Portugal, SGPS, SA, em resultado da reestruturação do sector energético em Portugal, para operar no sector petrolífero e do gás natural.

Actualmente, a Galp Energia é o principal grupo integrado de produtos petrolíferos e gás natural do país, com forte e crescente presença em Espanha e em actividade de desenvolvimento no sector de produção e de fornecimento de energia eléctrica [1]. As principais actividades do grupo são:

- Exploração e Produção;
- Refinação e Distribuição de Produtos Petrolíferos;
- Gás Natural
 - Aprovisionamento e Venda de Gás Natural
 - Distribuição de Gás Natural;
- Energia.

As principais empresas que integram o grupo Galp Energia, dedicadas as actividades acima mencionadas, são:

- Petrogal - dedicada à refinação de petróleo bruto e seus derivados, bem como ao seu transporte, distribuição e comercialização. É igualmente responsável pela pesquisa e exploração de petróleo bruto e gás natural;
- Galp Energia - empresa que gere as participações sociais de outras actividades do sector energético, como forma indirecta do exercício de actividades económicas;
- Galp Gás Natural - responsável pela comercialização de gás natural, no estado gasoso ou liquefeito, incluindo aprovisionamento e a sua colocação em território nacional ou no mercado internacional e,
- Lisboagás - empresa distribuidora de gás natural de média ou baixa pressão, na área geográfica de concessão, abrangendo designadamente a construção e operação de infra-estruturas que integrem a Rede Nacional de Distribuição de Gás Natural, a promoção da construção, conversão ou adequação de instalações de utilização de gás natural, entre outras actividades acessórias.

O sector da refinação incorpora duas refinarias estrategicamente localizadas em Matosinhos e Sines. A capacidade de processamento é de 4,8 e 10,4 Mton de petróleo bruto por ano, respectivamente, sendo responsáveis por colmatar cerca de 80% da procura em termos de produtos derivados do petróleo em Portugal e representam 20% da capacidade refinadora na Península Ibérica.

2.2. Refinaria de Matosinhos

A refinaria de Matosinhos da Petrogal, localizada no concelho de Matosinhos, possui uma área aproximada de 200 hectares e encontra-se interligada ao terminal petrolífero do porto de Leixões por oleodutos com cerca de 2 km de extensão. Na figura 2.1 apresenta-se uma imagem aérea da refinaria, onde se pode constatar que a maior extensão é ocupada pelo sector de armazenagem (depósitos circulares).



Figura 2.1 - Fotografia aérea da refinaria de Matosinhos [1].

A configuração processual da refinaria de Matosinhos permite a obtenção de diversos produtos:

- Combustíveis,
- Lubrificantes,
- Produtos aromáticos de base: benzeno, tolueno e xilenos e,
- Solventes industriais.

Estes diversos produtos provêm do funcionamento integrado das cinco fábricas que compõem o complexo industrial:

- Fábrica de Combustíveis (*FCO*) - produção de combustíveis: gasolinas e gasóleo,
- Fábrica de Óleos Base (*FOB*) - produção de óleos base para lubrificantes,
- Fábrica de Utilidades (*FUT*) - produção de energia eléctrica, vapor e água para utilização nas outras fábricas do complexo,
- Fábrica de Aromáticos (*FAR*) - produção de benzeno, tolueno e xilenos,
- Movimentação de Produtos (*MOV*) - armazenagem e *blending* (quando necessário) dos produtos provenientes das outras fábricas.

As águas residuais para tratamento são enviadas para a ETAR (Estação de Tratamento de Águas Residuais), onde sofrem diversos processos de tratamento para despoluir os diferentes efluentes que recebe e, posteriormente, efectuar a descarga controlada para o mar.

A tabela 2.1 mostra a capacidade de fabrico das diferentes linhas de produção, distribuídas pelas fábricas do complexo acima mencionadas.

Tabela 2.1 - Capacidades de fabrico da refinaria de Matosinhos [1].

Produto	Capacidade de fabrico (ton/ano)
Produção de combustíveis	3 700 000
Produção de óleos base	150 000
Produção de aromáticos e solventes	440 000
Fabricação de massas lubrificantes	1 500
Fabricação e modelação de parafinas	10 000
Produção de betumes	150 000
Produção de enxofre	10 000

A expedição dos produtos finais, que se podem encontrar num dos três estados físicos (sólido, líquido ou gasoso), é efectuada por uma de três vias:

- Via Rodoviária - os produtos líquidos são expedidos em veículos-cisterna ou embalados,
- Via Marítima - através de oleodutos para o porto de Leixões, onde são transportados para o cliente final por navio;
- Por oleoduto - abastecimento directo entre a refinaria e o cliente final que possui armazenagem própria, como é o caso do abastecimento de propano e butano para os Parques Real e de Perafita, ou o abastecimento de *jet* (combustível de aviação) para o aeroporto do Porto (Francisco Sá Carneiro).

Esta dissertação terá como alvo duas unidades processuais que fazem parte da fábrica de combustíveis: unidade de dessulfuração de gasóleo e unidade de conversão catalítica de sulfureto de hidrogénio.

2.3. Fábrica de Combustíveis

A fábrica de combustíveis (*FCO* - figura 2.2) é constituída por catorze unidades processuais, sendo a maior fábrica do complexo.

Os processos físico-químicos que ocorrem nesta fábrica são apresentados na figura 2.3, sendo que a partir do petróleo bruto é possível obter uma vasta gama de produtos combustíveis, incluindo grande parte da matéria-prima constituinte da corrente de alimentação a outras fábricas como a *FAR*.



Figura 2.2 - Localização da fábrica de combustíveis no complexo da refinaria de Matosinhos [1].

Devido à sua grande flexibilidade, esta fábrica permite o tratamento de diferentes tipos de petróleo bruto, desde petróleos mais leves aos mais pesados, sendo apenas necessárias alterações de interligações processuais e ajuste de variáveis operacionais.

As duas unidades processuais alvo desta dissertação encontram-se na figura 2.3, com as seguintes denominações: Dessulfuração de Gasóleo I e II e Recuperação de Enxofre. Nas duas secções seguintes serão abordadas estas duas unidades separadamente.

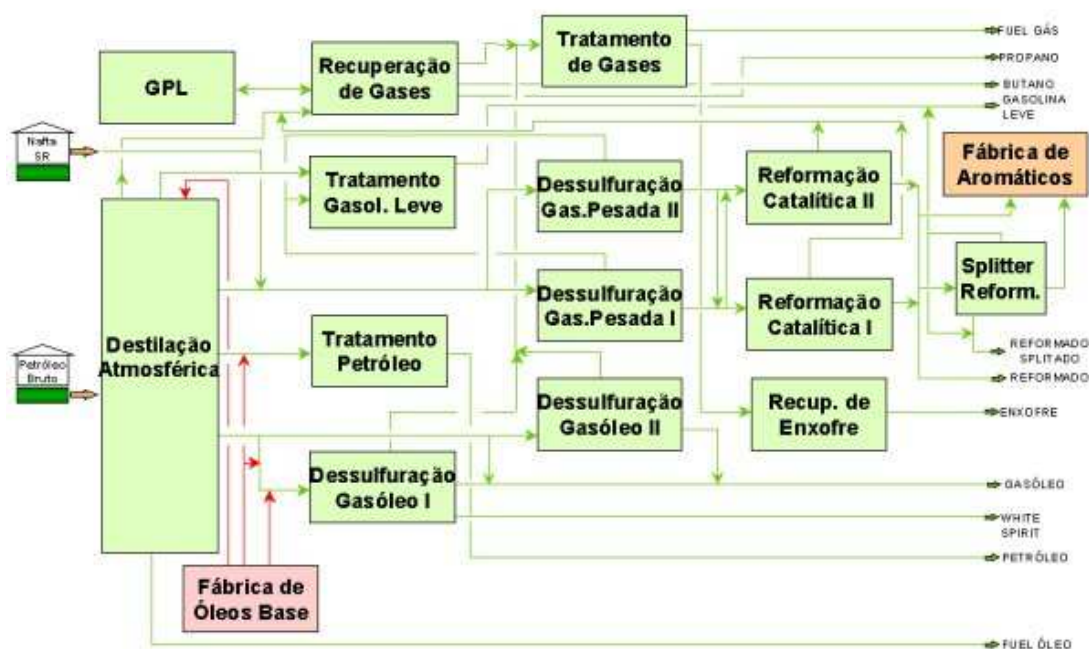


Figura 2.3 - Diagrama simplificado das unidades processuais da fábrica de combustíveis [1].

2.3.1. Unidade de Dessulfuração de Gasóleo - UN 3700

Esta unidade processual pertencente à fábrica de combustíveis, tem como objectivo reduzir o teor em enxofre no gasóleo utilizando diferentes processos físico-químicos, de forma a cumprir requisitos legais. A figura 2.4 apresenta um diagrama simplificado da unidade.

Trata-se dum processo de hidrodessulfuração do gasóleo (reação catalítica, realizada a alta pressão parcial de hidrogénio, na presença dum catalisador apropriado) que reduz drasticamente o seu teor de enxofre.

Em consequência da severidade da operação, formam-se alguns produtos leves por craqueamento⁽²⁾ da carga, que são separados por *stripping*⁽³⁾ do gasóleo tratado, antes de este ser enviado para a armazenagem.

² - Craqueamento- Processo de quebra de ligações químicas entre elementos constituintes de um dado composto, obtendo-se compostos mais leve do que o inicial

³ - *Stripping* - Método de separação de um ou mais componentes de uma fase líquida utilizando uma corrente gasosa

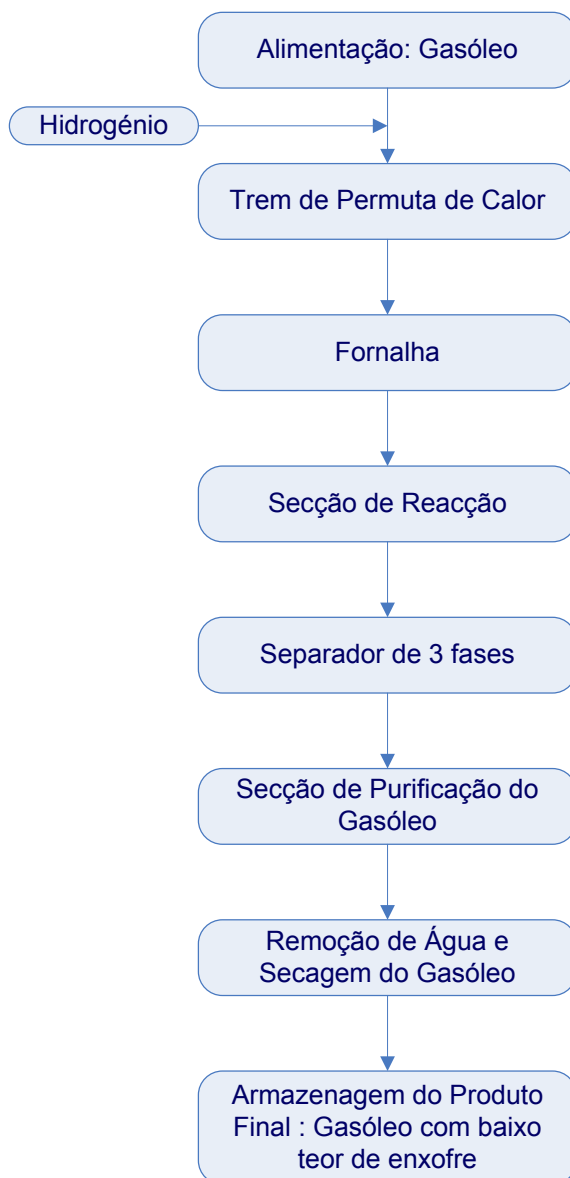


Figura 2.4 - Esquema simplificado da UN3700.

2.3.2. Unidade de Conversão Catalítica de Sulfureto de Hidrogénio - UN 3800

A unidade 3800 (figura 2.5) pertencente à fábrica de combustíveis converte o sulfureto de hidrogénio (H_2S) contido na corrente gasosa proveniente do Tratamento de Gases, em enxofre elementar (S).

Nesta unidade a corrente rica em H_2S é queimada num fornalha a alta temperatura (cerca de $1100^{\circ}C$), permitindo a formação de vários compostos de enxofre em que, aproximadamente 70% é enxofre elementar.

Esta corrente será arrefecida e algum enxofre condensa sendo separado por decantação. A restante corrente vai a um processo de reacção catalítica onde os restantes compostos serão convertidos em enxofre elementar. Este processo, designado por *Claus*, compreende a queima de um terço da corrente de H_2S para formação de dióxido de enxofre (SO_2) que, reagindo com os dois terços remanescentes, na presença de um catalisador, produz enxofre elementar.

Este é um processo importante da fileira de tratamento na medida em que recupera, como produto comercial, todo o enxofre removido dos diferentes produtos pelos respectivos processos de purificação, cumprindo requisitos ambientais impostos.

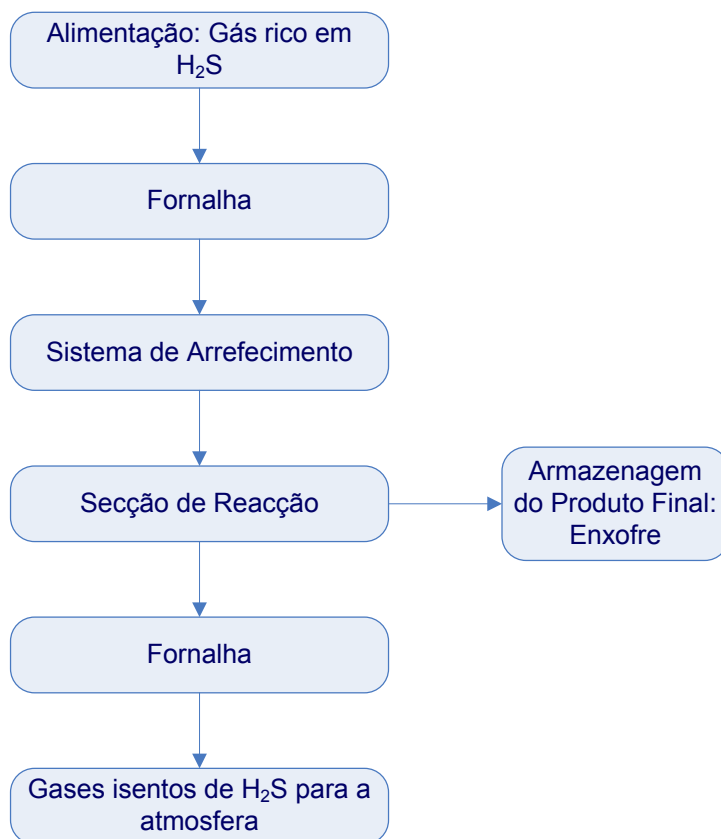


Figura 2.5 - Esquema simplificado da UN3800.

Capítulo 3

Sistemas de Controlo Distribuído

Como referido na Introdução, na refinaria de Matosinhos são utilizados simultaneamente dois sistemas de controlo distribuído, sendo a base de configuração e operação das diferentes unidades processuais. Assim sendo, torna-se fundamental abordar a temática dos sistemas de controlo distribuído de uma forma sistemática.

Neste capítulo pretendem-se apresentar as funcionalidades e características básicas de um sistema de controlo distribuído, bem como abordar de forma sucinta a evolução temporal dos mesmos, bem como a sua implementação na refinaria de Matosinhos.

3.1. Introdução

Os sistemas de controlo distribuído (*Distributed Control Systems - DCS*) são sistemas de controlo vocacionado para o controlo e supervisão das indústrias de processos contínuos. Estes sistemas surgiram, inicialmente, para substituir os painéis de controlo analógicos, tendo, posteriormente, adquirido funcionalidades como o controlo sequencial de processos tipo *batch* [2].

De forma a clarificar a definição de um *DCS* apresentam-se as figuras 3.1 a 3.3. A primeira mostra de uma forma esquemática um sistema de controlo centralizado, onde a medição processual (realizada através do sensor) é enviada para um único controlador (através de uma ligação ponto-a-ponto) que, por sua vez, determina o valor de saída para o actuador local.

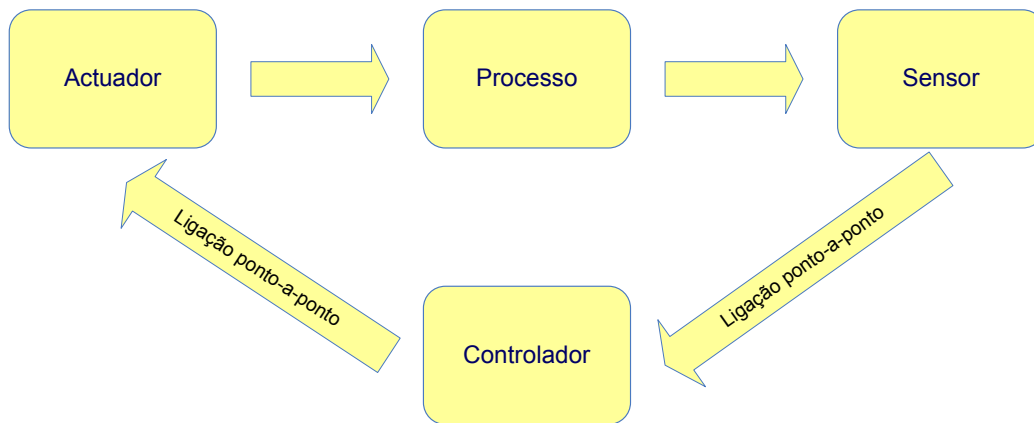


Figura 3.1 - Sistema de controlo centralizado [3].

A deslocalização do nó de controlo é explícita na segunda figura, em que a ligação entre o sensor/actuador é efectuada utilizando uma rede de comunicação sendo, assim, descentralizado. Neste caso, o algoritmo de controlo é executado num único controlador mas algumas tarefas mais simples (p.e. condicionamento de sinal) podem ser executadas a montante deste.

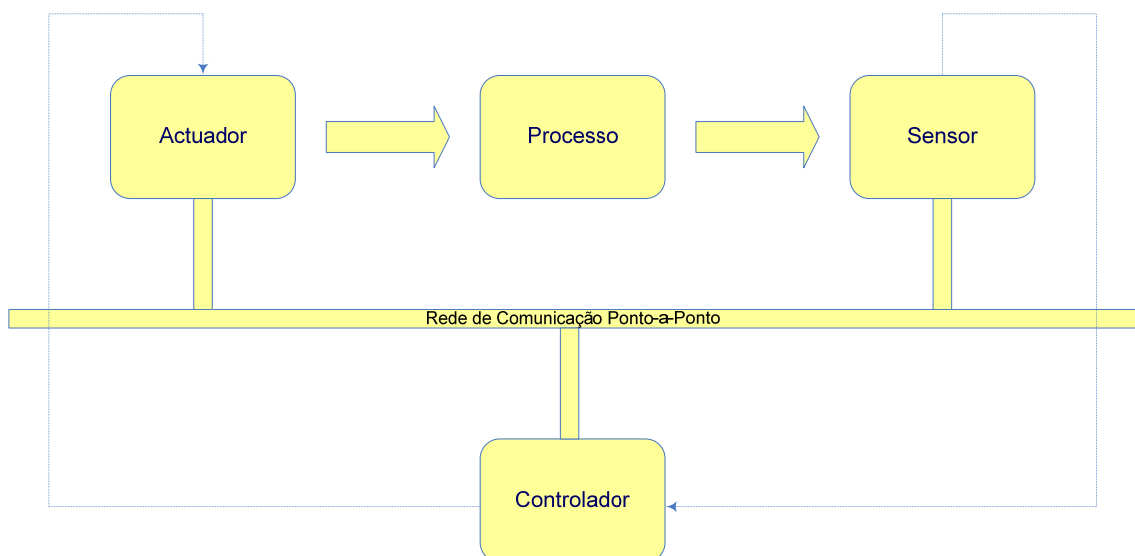


Figura 3.2 - Sistema de controlo descentralizado [3].

Num sistema de controlo distribuído (figura 3.3) existem vários controladores ligados numa rede de comunicação mais complexa. Os controladores trocam dados entre si, cumprindo o objectivo de interligar os equipamentos de menor complexidade (sensores e actuadores) aos de maior complexidade. Os algoritmos de controlo estão distribuídos pelos diversos controladores que compõe o sistema. Para estes sistemas a rede de comunicação tem que ser extremamente fiável, sendo muitas vezes redundante, de forma a minimizar eventuais perturbações processuais por perdas de comunicação.

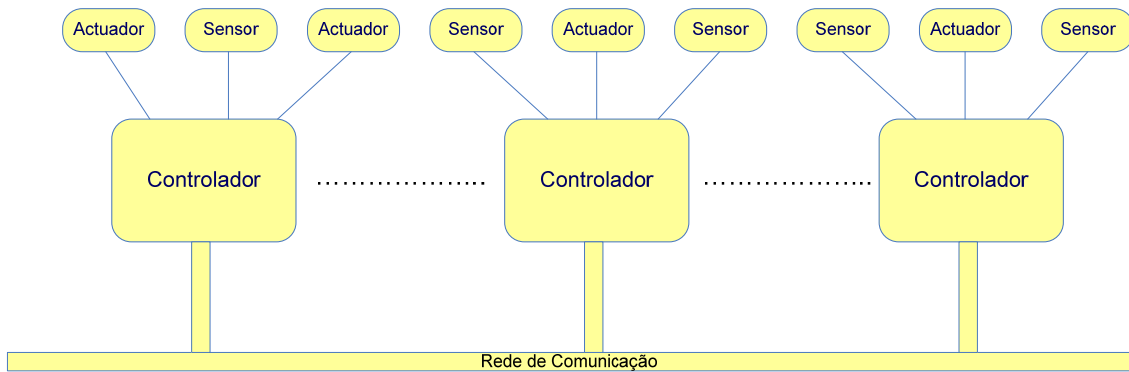


Figura 3.3 - Sistema de controlo distribuído.

3.2. Evolução Histórica

A utilização do controlo digital (*DDC - Direct Digital Control*) em painéis que se estendiam por longos metros era o mais comum em processos contínuos até à década de 50 [4]. A introdução dos microprocessadores na indústria nos anos 60 marcou o início de uma revolução ao nível do controlo de processos. Em 1970 a *Taylor Instrument Company* [5] e a *Foxboro* [6] lançaram os sistemas *1010* e *FOX1*, respectivamente. Apesar de serem *DDC* já se baseavam na utilização de mini-computadores e possuíam sinais de *E/S* (Entrada/Saída) ligados a *hardware* proprietário.

A 30 de Novembro de 1964 a *IBM* lançou o controlador *IBM 1800* [7], com capacidade de aquisição de dados e controlo processual, permitindo controlar centenas de variáveis processuais à velocidade de 8 Mb/s, sendo um sistema centralizado. Apesar deste sistema ser obsoleto nos dias de hoje existe uma empresa no Canadá ligada à energia nuclear que o preserva em pleno funcionamento, figura 3.4.



Figura 3.4 - IBM 1800: Aquisição de dados e sistema de controlo centralizado [7].

A partir dos anos 70 várias empresas lançaram para o mercado os primeiros DCS baseados em soluções de *hardware* e *software* proprietárias. Em 1975 a *Honeywell* apresentou o *TDC 2000* [8] (fotografia dos controladores na figura 3.5), a *Yokogawa* surge com o *CENTUM* [9] (exemplo do *HMI* - *Human Machine Interface* - na figura 3.6) e a *Bristol* com o controlador *UCS 3000*.

Em 1980 a *Bailey*, actualmente pertencente à *ABB*, lançou o *NETWORK 90* e, nesse mesmo ano, a *Fisher & Porter Company* (também adquirida pela *ABB*) apresentou o *DCI-4000*. A sigla *DCI* significa *Distributed Control Instrumentation* e realça a importância da descentralização do sistema de controlo.

A ligação de controladores, estações de trabalho e outros elementos ligados a uma rede de comunicação constituía uma vantagem competitiva para as empresas que apostaram neste tipo de soluções. No entanto, era necessária uma rede de comunicação extremamente fiável e, assim, a redundância da rede de comunicação começou a fazer parte de soluções existentes no mercado.



Figura 3.5 - Controladores do *DCS TDC 2000* da *Honeywell* [8].



Figura 3.6 - *HMI* do *DCS CENTUM* da *Yokogawa* [9].

Os *DCS* caracterizavam-se por serem robustos ao nível de *hardware* mas complexos ao nível de configuração/*software* e, portanto, necessitavam de técnicos altamente qualificados para a montagem e configuração do sistema. Outra das desvantagens apresentadas relacionava-se com a impossibilidade de troca de informação com outros sistemas, ou seja,

apenas era possível obter informação utilizando impressoras ligadas directamente ao *HMI* ou ao barramento de comunicação.

Foi a partir do sistema operativo *UNIX* e do protocolo de comunicação *TCP/IP* que se iniciou a abertura dos *DCS* ao mundo da partilha de informação. Nos anos 80 foi efectuada a primeira ligação de um *PLC* a um *DCS*. A *Foxboro* foi a empresa que pela primeira vez aliou o *UNIX* à rede *Ethernet* quando introduziu o sistema *I/A* em 1987 [10].

Com as constantes evoluções ao nível de sistemas operativos e tendo a *Microsoft* o monopólio da sua implementação em ambiente industrial (em estações de trabalho e servidores), as redes de comunicação apresentavam cada vez maior flexibilidade e possibilidade de integração de dados com outros sistemas. No caso da *Yokogawa* (figura 3.7) pode-se acompanhar as sucessivas evoluções do *CENTUM* desde a sua primeira versão até à última lançada em 2008: *CENTUM VP*. Desde 1993 que esta empresa aposta na utilização de barramento de comunicação *Ethernet* [9].

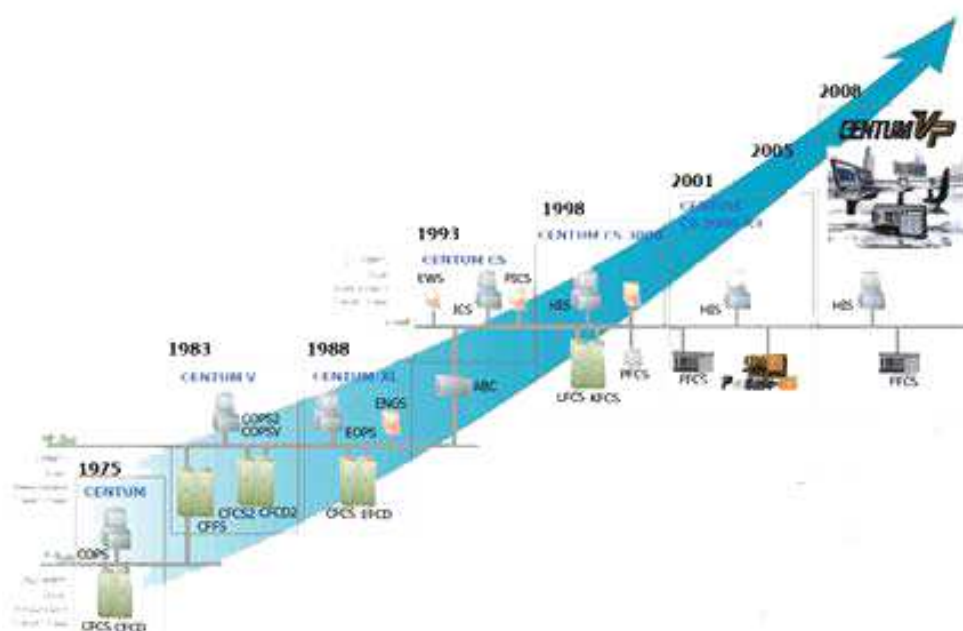


Figura 3.7 - Evolução dos *DCS* da *Yokogawa* desde o lançamento do *CENTUM* em 1975 até ao *CENTUM VP* em 2008 [9].

A partir dos anos 90, a importância crescente do sistema de armazenamento de dados para histórico, da construção de tendências de variáveis processuais, da flexibilização da base de dados, da interligação com sistemas externos (utilização de *OPC OLE for Process Control*) e de outras ferramentas que são acrescentadas como mais-valias, como o sistema de gestão de alarmes, sintonização de controladores, ferramentas de manutenção predictiva ou comunicação com diferentes redes de campo como o *FOUNDATION™ fieldbus*, imprimiram aos *DCS* uma nova dinâmica. Estes tendem a tornarem-se cada vez mais completos, podendo mesmo integrar lógicas de seguranças processuais que são habitualmente realizadas em *PLC*.

Devido à redução dos custos de implementação de um *DCS*, é cada vez mais frequente a opção por este tipo de sistema em detrimento de um sistema de controlo isolado. Actualmente, as empresas apostam no aumento de capacidade do sistema ao nível de *E/S*

bem como na capacidade de processamento da informação com controladores com maior memória e de menores dimensões.

3.3. Arquitectura

A arquitectura de um sistema de controlo distribuído pode ser dividida em duas partes distintas mas integradas: *hardware* e *software*.

Ao nível da arquitectura de *hardware* os DCS são constituídos por controladores que processam a informação proveniente de cartas de *E/S* e um barramento que permite a comunicação dessa informação para estações de trabalho e servidores. Habitualmente o barramento de comunicação é redundante tal como os controladores, embora seja sempre uma opção do gestor do sistema. As cartas de *E/S* podem igualmente ser redundantes tal como as estações de trabalho.

Diferentes tipos de estações de trabalho poderão coexistir no sistema, embora com limites bem definidos ao nível de quantidade: estações de configuração, onde poderá ser configurado o *hardware* e respectivas estratégias de controlo; estações de armazenamento de dados, sendo responsáveis por fornecer dados processuais históricos para a correcta operação das unidades processuais; estações de operações (*HMI*); estação de manutenção, que permitem efectuar diagnósticos e recolha de informação relativa a equipamento de campo como válvulas de controlo, instrumentação inteligente ou com a tecnologia *FOUNDATION™ fieldbus*.

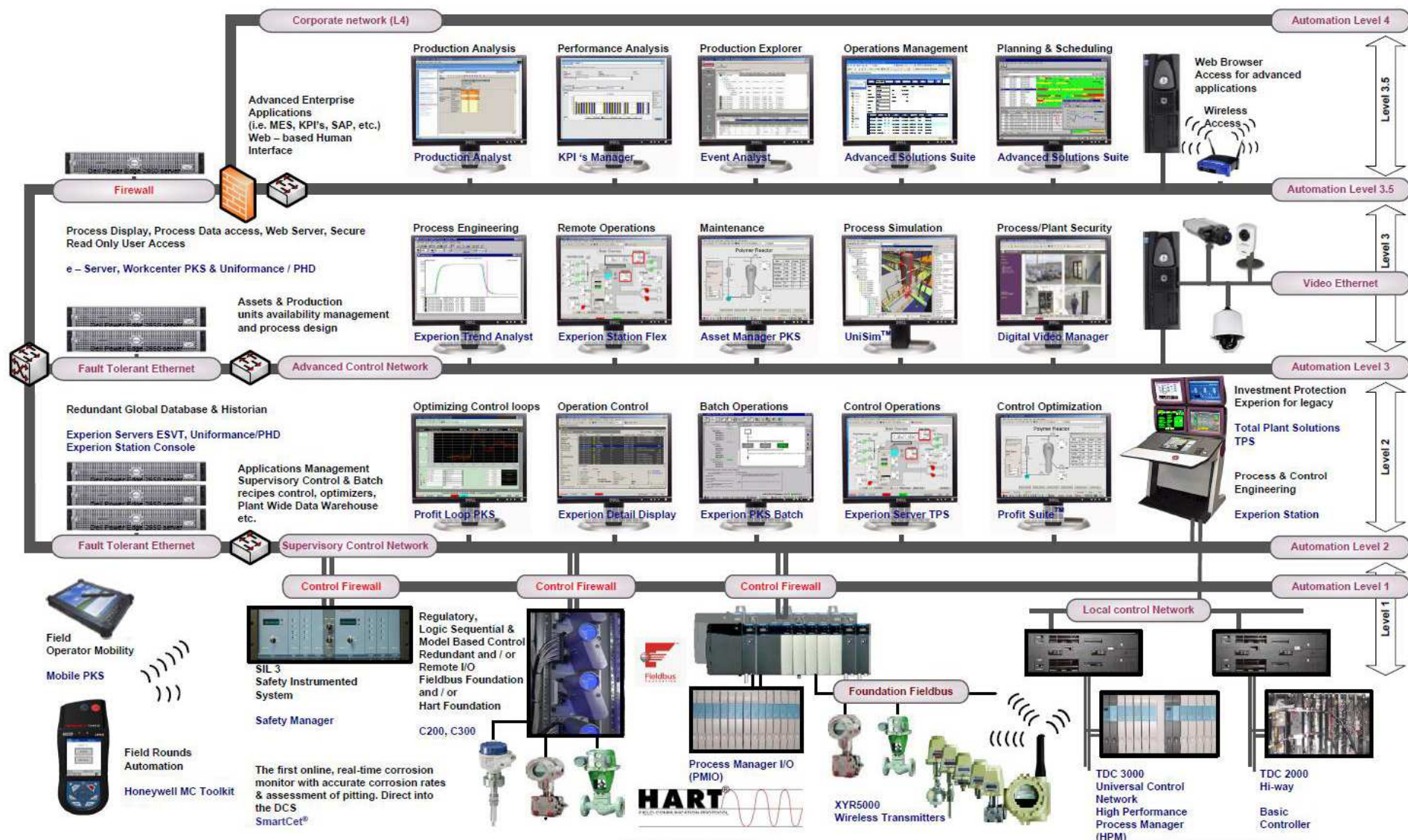


Figura 3.8 - Arquitetura física ou de hardware do sistema *EXPERION PKS* da *Honeywell* [11].

Como exemplo, é apresentada, na figura 3.8, a arquitectura física do sistema de controlo distribuído *EXPERION Process Knowledge System (PKS)* da *Honeywell*. A estrutura encontra-se dividida em diferentes níveis desde o nível de campo, nível 1, até à rede empresarial, nível 4. Nesta arquitectura, bem como nas demais existentes no mercado, o nível posterior comporta, obrigatoriamente, informação do nível que o antecede.

O *EXPERION PKS*, no nível 1, permite a comunicação com instrumentação e/ou equipamento de campo usando redes de campo como o *FOUNDATION™ fieldbus*, *HART*, *Profibus*, *ControlNet*, entre outras, sendo também possível a integração com *DCS* obsoletos ou que estão prestes a deixarem de ser comercializados, como o *TDC 2000* e o *TDC 3000*. No nível seguinte, e com a integração de todas as estações de trabalho, a rede de supervisão fica completa, integrando a informação proveniente do nível anterior. Em diversas sistemas de controlo os níveis 3 e 4 são inexistentes, ou seja, o sistema poderia funcionar apenas com os dois primeiros níveis, embora o grau de integração fosse diminuto, permitindo apenas o controlo e acesso local à informação disponibilizada.

O nível 3 prende-se com a utilização de ferramentas avançadas de optimização processual ou de manutenção. Habitualmente, são usadas ferramentas acessórias para sintonização de malhas de controlo, manutenção predictiva de equipamentos críticos ou semi-críticos, simulação processual e formação de operadores, entre outros. Alguns *DCS* já oferecem algumas destas ferramentas embebidas no seu próprio *software*, sendo apenas necessário a aquisição de licenças adicionais.

A integração da rede processual/controlo com a rede empresarial é, usualmente, precedida por uma *firewall* que, tem como principal objectivo, proteger a rede processual/controlo de ameaças de vírus e entradas de *hackers* ou curiosos. O nível 4 permite, assim, o acesso à informação tratada ou não, que possibilitam, por exemplo, o fecho do balanço material e energético às unidades processuais ou a programação das produções para um determinado período temporal previamente definido.

Ao nível de *software* foi já referida a importância da utilização de computadores pessoais com sistemas operativos que permitem obter interfaces muito mais atractivas para o utilizador, incorporação de folhas de cálculo que podem interagir directamente com bases de dados, diagnósticos avançados da rede de controlo, troca de informação com *softwares* de gestão ou com bases de dados externas, entre outras. As ferramentas avançadas de optimização processual, manutenção predictiva, simulação e formação de operadores são, igualmente, suportadas pelos sistemas operativos tornando a sua utilização quase intuitiva e o ambiente gráfico muito similar entre as aplicações.

3.4. Aplicações Típicas de um DCS

Os *DCS* são, actualmente, de uso generalizado na indústria dos processos contínuos e, devido às constantes evoluções que têm sofrido ao longo da última década, também são utilizados no controlo de processos *batch*.

É na indústria do papel, química, refinação e farmacêutica que mais frequentemente se podem encontrar *DCS* devido ao elevado número de *E/S* e complexidade de controlo dos processos físico-químicos. O elevado custo de implementação de um *DCS* só pode ser comportado quando a instalação o justifica, nomeadamente pelos dois aspectos referidos anteriormente: elevado número de *E/S* ou devido a complexidade processual.

A utilização de *PLC* com funcionalidades semelhantes a um *DCS* tem vindo a aumentar contudo, os primeiros são utilizados em processos rápidos como controlo de equipamentos (motores ou turbinas), e os *DCS* são mais vocacionados para processos contínuos de alguma complexidade. Para ultrapassar este problema os fabricantes de *DCS* conceberam soluções para processos de pequena dimensão que fossem competitivos em relação aos *PLC*.

A competitividade entre os fabricantes de *PLC* e *DCS* conduziu ao aparecimento de soluções híbridas, captando o melhor de cada um destes sistemas. Um exemplo é o controlador programável para automação (*PAC - Programmable Automation Controller*) que permite ter as funcionalidades de controlo analógico de um *DCS* acompanhada pela elevada capacidade de controlo discretos de um *PLC*. O gráfico da figura 3.9 mostra que os *PAC* possuem a fiabilidade de um *PLC* e as funcionalidades de um *PC* (*Personal Computer*).

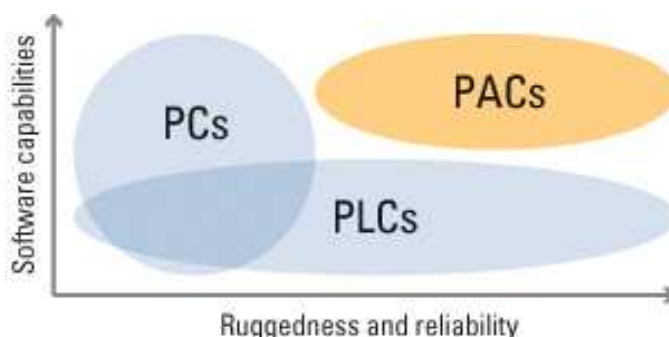


Figura 3.9 - Fiabilidade vs. funcionalidade de *PC*, *PLC* e *PAC* [13].

Apesar desta solução híbrida ser atractiva, as indústrias de refinação e químicas continuam a preferir a utilização de *DCS*, visto que, sempre que for necessário, é possível total integração de soluções baseadas em *PLC*. Assim sendo, os *PLC* nestas indústrias continuam a ser vastamente utilizados para funções específicas como controlo de motores, sistemas de segurança processual, ou controlo de pequenas unidades que são vendidas como um pacote de equipamento e lógica de controlo.

A opção para a modernização dos sistemas de controlo tem passado pela selecção do *DCS* mais moderno da mesma marca que o anterior, garantindo assim perfeita integração da nova solução, permitindo também minimizar o tempo de paragem de unidades processuais para efeitos de migração do sistema de controlo, pois implicam elevadas perdas económicas.

Capítulo 4

Sistemas de Controlo Distribuído na Refinaria de Matosinhos

Neste capítulo serão descritos, com detalhe, os dois sistemas de controlo distribuído implementados na refinaria de Matosinhos: o *RS3* e o *DeltaV*.

A primeira parte será dedicada ao *RS3* e a segunda ao *DeltaV*, seguindo assim a cronologia da introdução dos sistemas na refinaria. A descrição dos sistemas passará pela arquitectura física e os diferentes componentes, e arquitectura do *software* e funcionalidades.

4.1. Rosemount System 3 - RS3

O sistema de controlo distribuído *RS3* concebido pela *Fisher Rosemount Systems*, agora *Emerson Process Mangement*, é baseado num barramento de comunicação, *hardware* e *software* proprietários.

Os principais componentes do sistema são os controladores designados por *ControlFiles*, responsáveis pelo processamento da informação e lógica, a interface com o operador/configurador, também denominadas por consolas ou *MTCC* (*Multitube Command Console*) onde são efectuadas as configurações dos *E/S*, lógica e operação processual, entre outros componentes de *hardware* para fins específicos, como interfaces para outros barramentos de comunicação.

Todos estes componentes são integrados num barramento de comunicações redundante e, também ele proprietário, o *PeerWay* (*PW*). A este barramento são ligadas todos os elementos de *hardware* directa, como o caso das *ControlFiles* e das *MTCC*, ou indirectamente, como no caso das cartas *E/S*, que se ligam às *ControlFiles*.

De forma a clarificar a arquitectura do *RS3*, são apresentados na figura 4.1 os diferentes elementos e a sua ligação ao barramento de comunicação. De notar que a ligação ao

barramento é efectuada recorrendo às *PeerWay Tap* que se vão ligando entre si, permitindo obter um sistema completamente integrado.

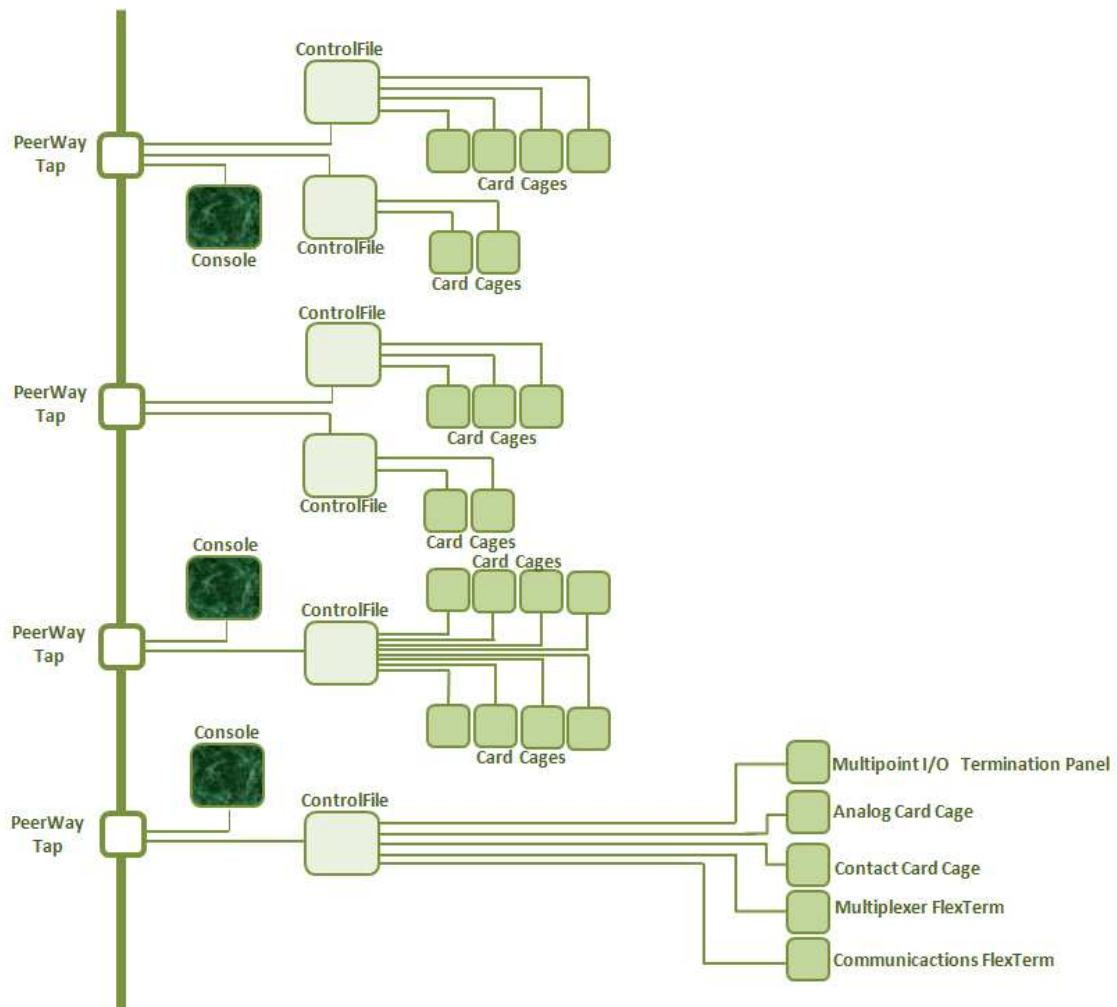


Figura 4.1 - Arquitectura do Sistema RS3 [14].

4.1.1. *PeerWay*

O barramento de comunicação redundante *PeerWay*, onde são interligados os elementos de *hardware*, permite a comunicação directa entre todos os dispositivos a ele ligados. É composto por dois cabos redundantes que podem ser *twinaxial* (*twinax* - um cabo coaxial com duas almas condutoras), fibra óptica ou uma combinação das duas.

Cada *PeerWay* pode comportar um máximo de 32 dispositivos ou nós. Um nó neste barramento pode ser uma *ControlFile*, uma *MTCC*, entre outros dispositivos de interface. De forma a interligar diferentes *PeerWays* são utilizados conectores designados de *HIA* (*Highway Interface Adapter*) - figura 4.2. Esta interligação é limitada a 31 *PeerWays*. Segundo dados da *Emerson Process Management* este limite nunca foi ultrapassado por nenhum complexo industrial.

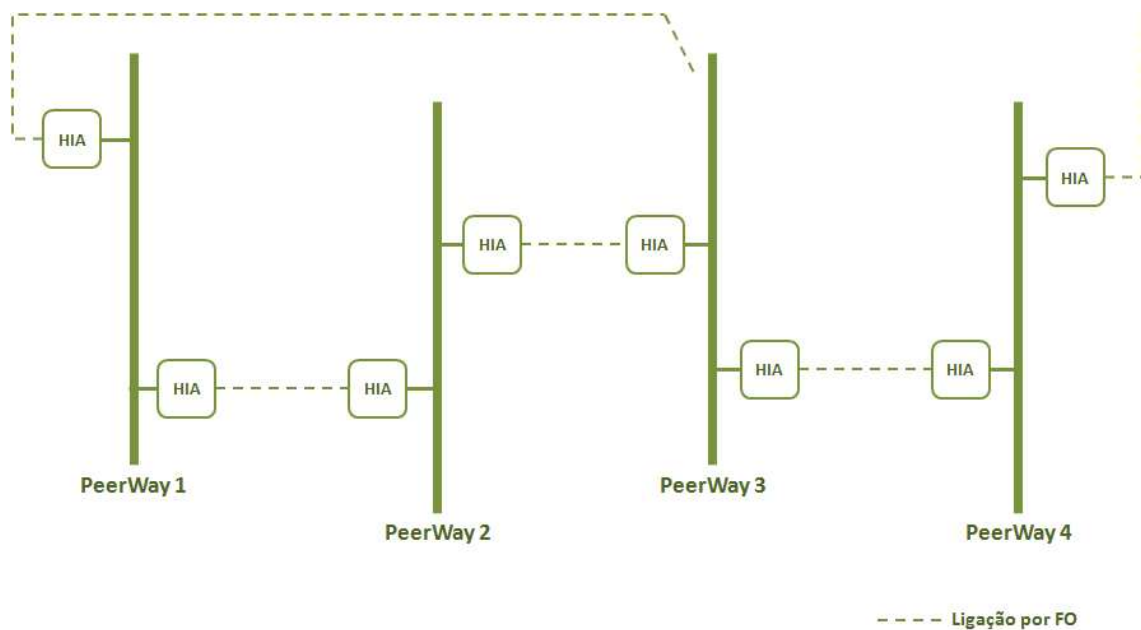


Figura 4.2 - Comunicação entre *PeerWays* via *HIA* [14].

A robustez do sistema é garantida pela utilização de componentes redundantes tanto ao nível do barramento de comunicações, como foi anteriormente referido, como também ao nível de *PeerWay Taps* que trabalham igualmente aos pares: *Tap A* e *Tap B* - figura 4.3. Cada nó do sistema é ligado a cada uma das *Taps* (A e B) que, por sua vez, estão ligadas ao *PeerWay* redundante (A e B). Cada *Tap* pode comportar no máximo 4 dispositivos. Esta configuração permite que, em caso de avaria de um dos componentes ou até mesmo perda de comunicação de um dos *PeerWays* o outro assuma toda a comunicação sem qualquer perturbação processual. Em funcionamento normal a comunicação encontra-se repartida equitativamente pelos dois barramentos.

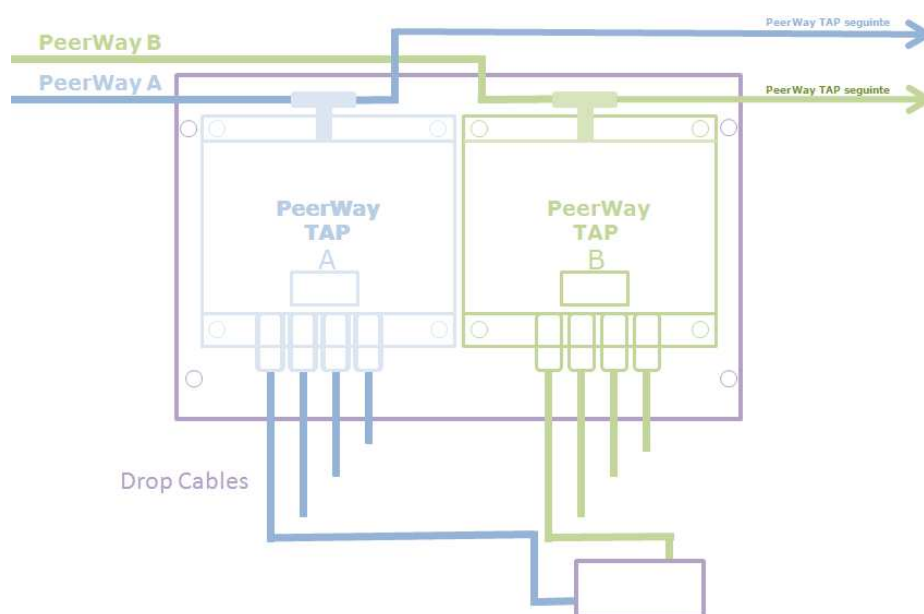


Figura 4.3 - *PeerWay Taps* redundantes [15].

Na figura 4.4 é apresentada a forma como os diferentes nós se ligam às *PeerWay Taps* e estas ao barramento *PeerWay* propriamente dito.

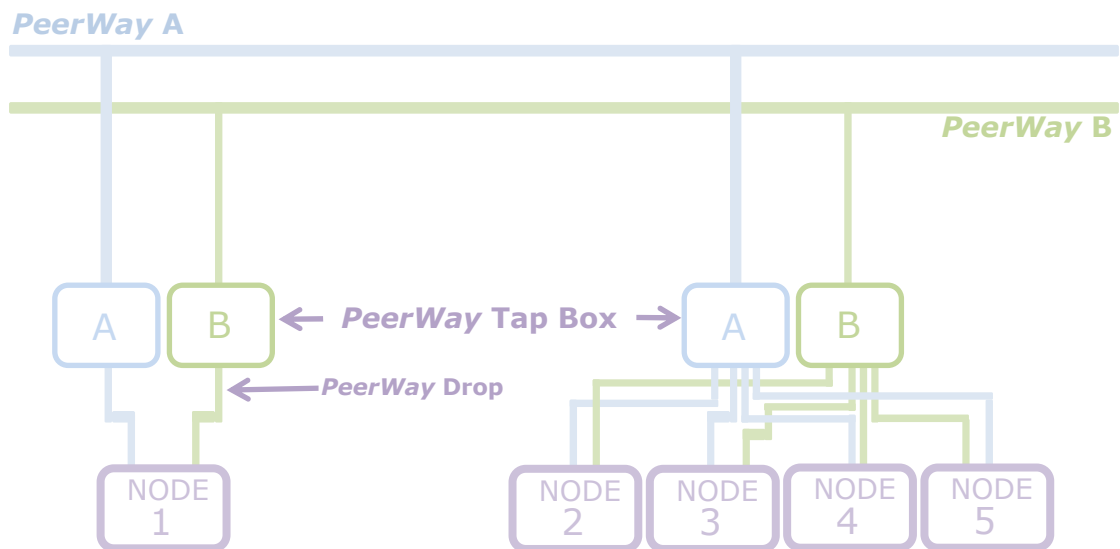


Figura 4.4 - Nós ligados às *PeerWay Tap* redundantes e ligação ao barramento *PeerWay* [14].

4.1.2. ControlFiles

As *ControlFiles* são um conjunto de cartas que têm como objectivo a monitorização dos *loops* de controlo e o processamento de toda a lógica implementada, permitindo também o envio de dados para outros equipamentos. Habitualmente, dividem-se as cartas constituintes das *ControlFile* em dois tipos (ver figura 4.5): cartas processadoras e cartas de suporte.

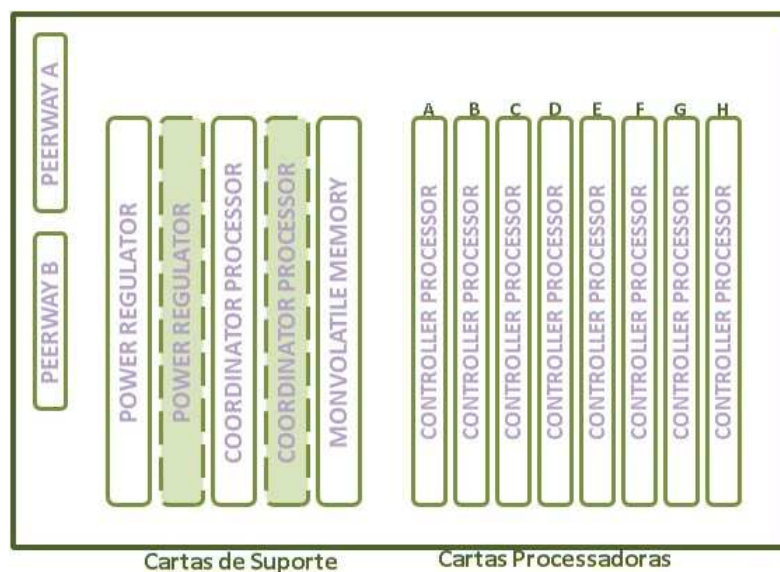


Figura 4.5 - Cartas constituintes da *ControlFile* [14].

As primeiras são denominadas por *MultiPurpose Controller Processor - MPC* e são responsáveis pela recepção e tratamento dos valores de entrada e por gerar valores de saída. As ligações de campo e respectivo condicionamento de sinal são efectuadas nas *Card Cages*, *Termination Panels*, *Flex Terms* e pelos *Marshalling Panels*, que comunicam com as *MPC* através de uma interface *RS-422*. Todos os valores que a carta *MPC* recebe ou gera podem ser disponibilizados para o *PeerWay* de forma a serem utilizados por outros nós. O processamento pode ser simples, quando é apenas necessário adquirir um valor de entrada, ou bastante complexo quando são necessários vários cálculos e passos lógicos para actualizar um ou vários valores de saída. Todo este processamento é executado pelas oito *MPC* (de *A* a *H* na figura 4.5) que podem ser ou não redundantes. O estado das cartas é verificado continuamente de forma a permitir a monitorização da entrada em funcionamento da carta em *standby*. Esta comutação é realizada num intervalo de tempo curto (1 s) de forma a não afectar o processo.

A comunicação entre as cartas processadoras e as cartas de interface de campo é controlada por cada carta processadora e não por cartas de suporte.

As cartas de suporte dedicadas às *MPC* são:

- *PeerWay Buffer*,
- *Power Regulator*,
- *Coordinator Processor*,
- *Nonvolatil Memory*.

As cartas *PeerWay Buffer* são responsáveis por manter um *buffer* entre o barramento de comunicação *PeerWay* e a *ControlFile*, bem como proporcionar o formato de comunicação entre eles. É igualmente esta carta que condiciona todas as comunicações entre os *PeerWays* e a carta *Coordinator Processor*, a qual será descrita mais adiante.

A carta *Power Regulator* permite a alimentação em corrente contínua e estabilizada (5V DC) para todas as cartas da *ControlFile*. Esta carta recebe a sua alimentação a partir de um barramento de 32 V DC que por sua vez poderá ser alimentado a partir de baterias ou de fontes de alimentação entre 19 e 36 V DC.

A carta *Coordinator Processor (CP)* gere as comunicações entre as oito possíveis cartas processadoras da *ControlFile*, bem como entre a *ControlFile* e o *PeerWay*. A actualização da base de dados na memória não-volátil com dados operacionais e configurações actualizadas, bem como o controlo de *downloads* necessários da memória não-volátil para as cartas processadoras são também da responsabilidade da carta *CP*.

A carta de memória não-volátil armazena o *software* das cartas processadoras, a configuração e dados operacionais de todas as cartas da *ControlFile* e também o *software* da *CP*. No caso de perda de configuração da carta processadora a base de dados será enviada pela *CP* a partir da memória não-volátil. Esta carta armazena a biblioteca de algoritmos do sistema de controlo *RS3*. Uma cópia desta biblioteca é enviada para as cartas processadoras quando são colocadas pela primeira vez em funcionamento.

A interacção das diversas cartas de suporte encontra-se esquematizada na figura 4.6.

Visto existirem dois *PeerWays* (*A* e *B*) existem duas cartas de *buffer*, duas *CP* e duas *Power Regulator* de forma a permitir total redundância.

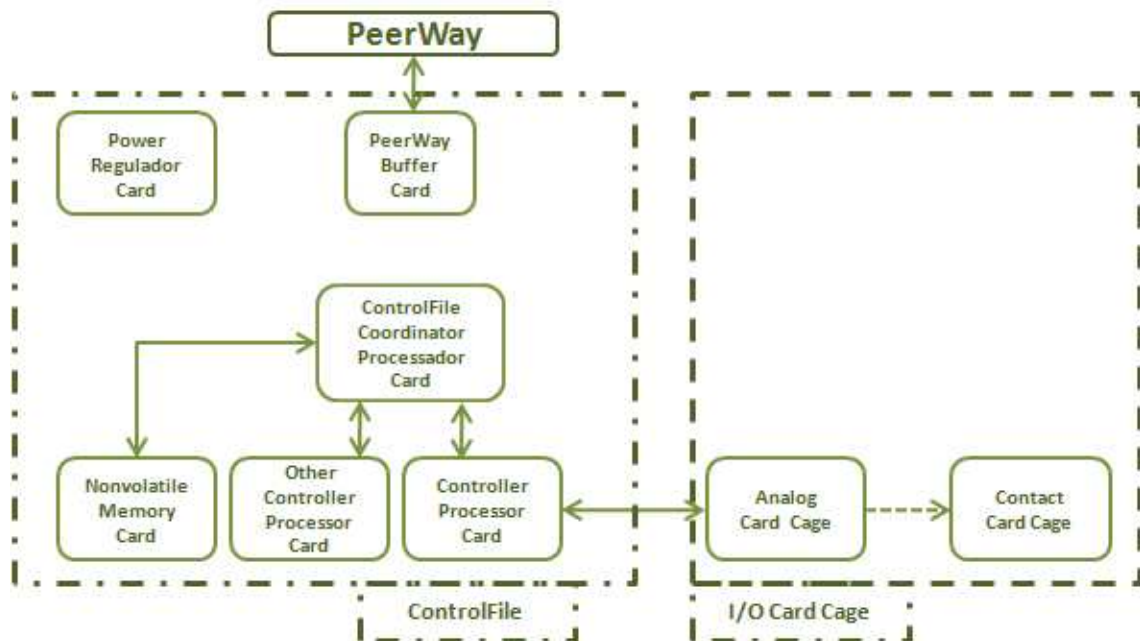


Figura 4.6 - Interação funcional entre as cartas de suporte das MPC [15].

4.1.3. Cartas de E/S

Ao nível de cartas de E/S o RS3 sofreu, ao longo da sua existência, diversas actualizações. Pretende-se nesta secção referir apenas os tipos de cartas utilizadas na refinaria de Matosinhos nos últimos cinco anos:

- *Multipoint E/S termination panels* para E/S discretos e analógicos,
- *Field Interface Cards - FIC* suportadas pelas *Analog card cages* para sinais analógicos contínuos ou por pulsos,
- *PLC cards* suportadas em *PLC card cages* para comunicação MODBUS com PLC. Existem três tipos de cartas dependendo das características de *pinout* da comunicação: RS422/RS232C, RS422/RS422 e RS422/485.

A figura 4.7 mostra a integração das cartas E/S e a respectiva ligação à *ControlFile* e *PeerWay*.

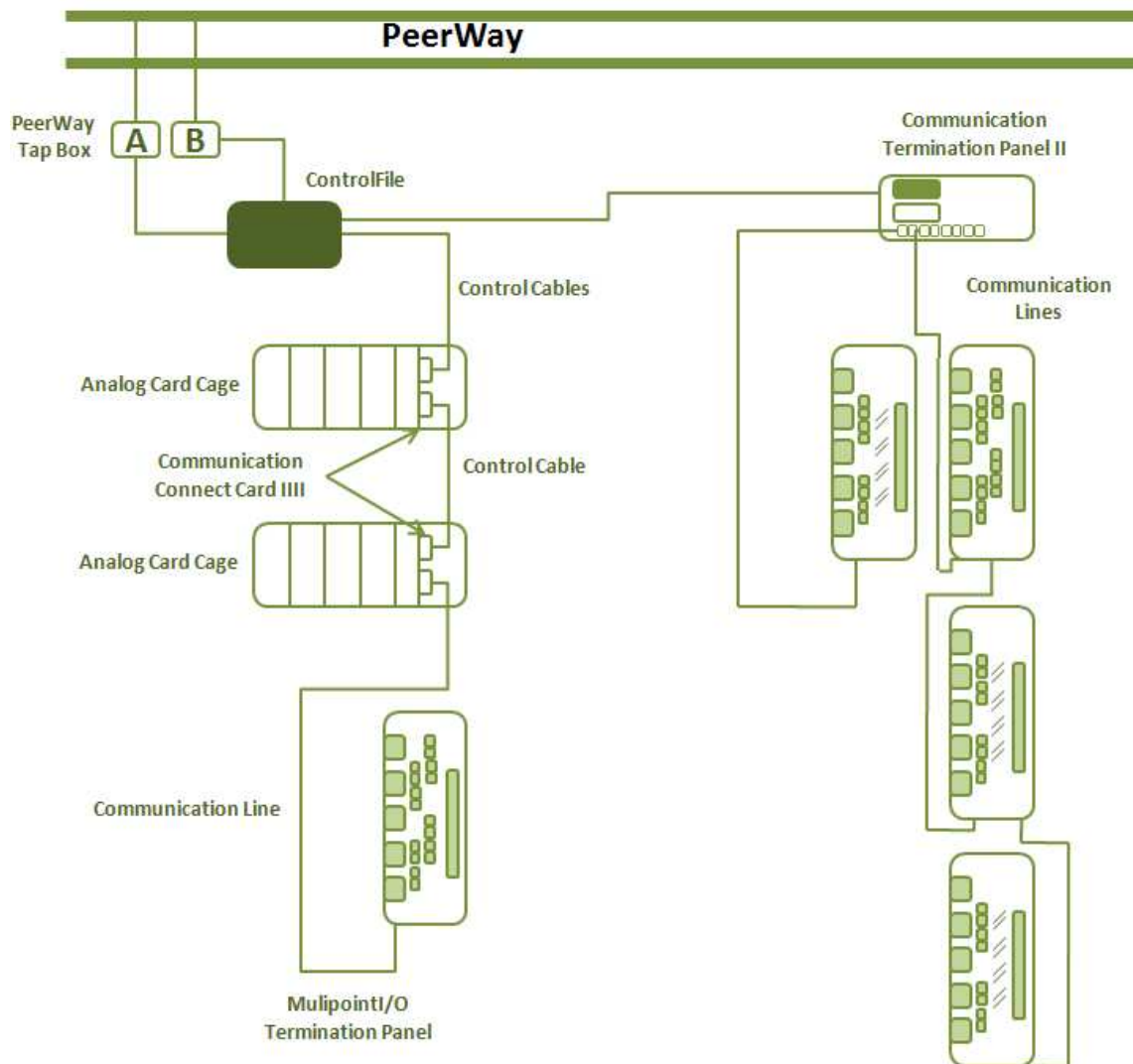


Figura 4.7 - Ligação das cartas E/S à ControlFile e ao PeerWay [14].

4.1.4. Consolas de Operação e Configuração

No RS3 existe apenas um tipo de consola que pode ser utilizada para operação e configuração: a *MTCC - Multitube Command Console*.

De forma a permitir controlo de acesso ao sistema é possível bloquear acessos ao nível de *software* e ao nível físico, este último utilizando uma chave física que é introduzida na consola e que, consoante o tipo de chave, pode permitir operar, supervisionar ou efectuar configurações no sistema. Ao nível de *software* podem ser bloqueados acessos por fábrica/área/unidade.

Nas consolas, o operador poderá operar a unidade recorrendo a *faceplates* (figura 4.8). Os *faceplates* são *standard* para alguns tipos de blocos e configuráveis para outros, permitindo alguma flexibilidade na operação.

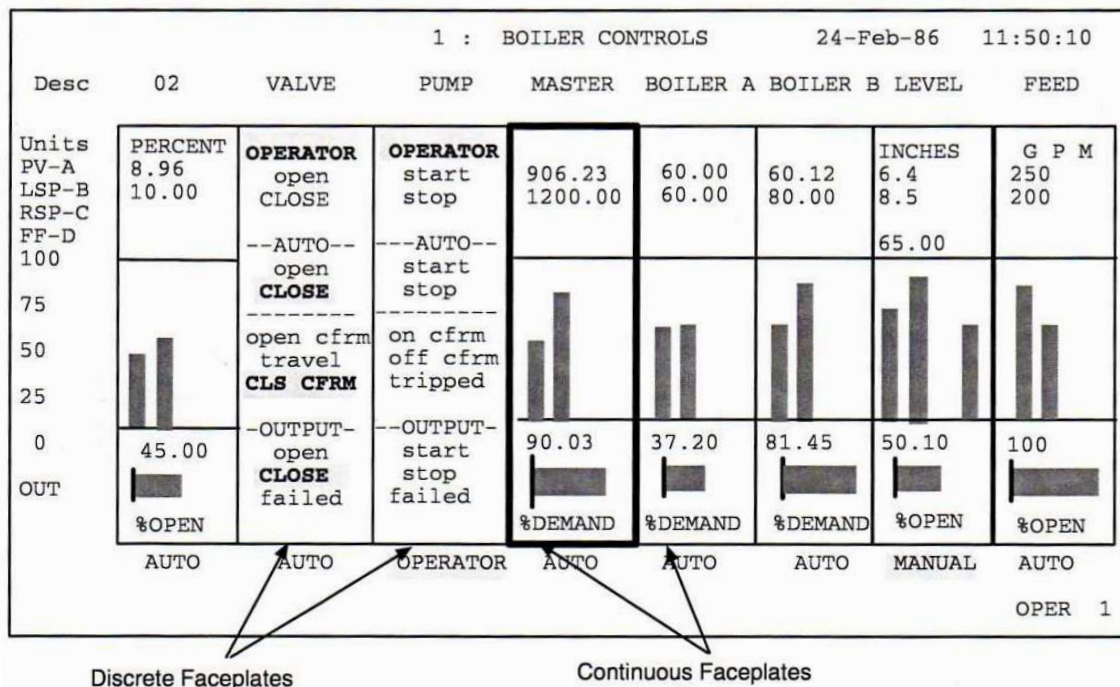


Figura 4.8 - Faceplates discretos e contínuos do RS3 [14].

Um auxiliar importante na condução das unidades processuais são as tendências, que não são mais do que gráficos onde é apresentada a curva relativa ao comportamento de uma dada variável ao longo do tempo (figura 4.9). Os dados relativos às variáveis podem ser gravados na própria consola ou noutra consola do *PeerWay*. O último caso deve ser evitado devido à saturação do barramento de comunicação e ao tempo excessivo de espera para visualizar a tendência.

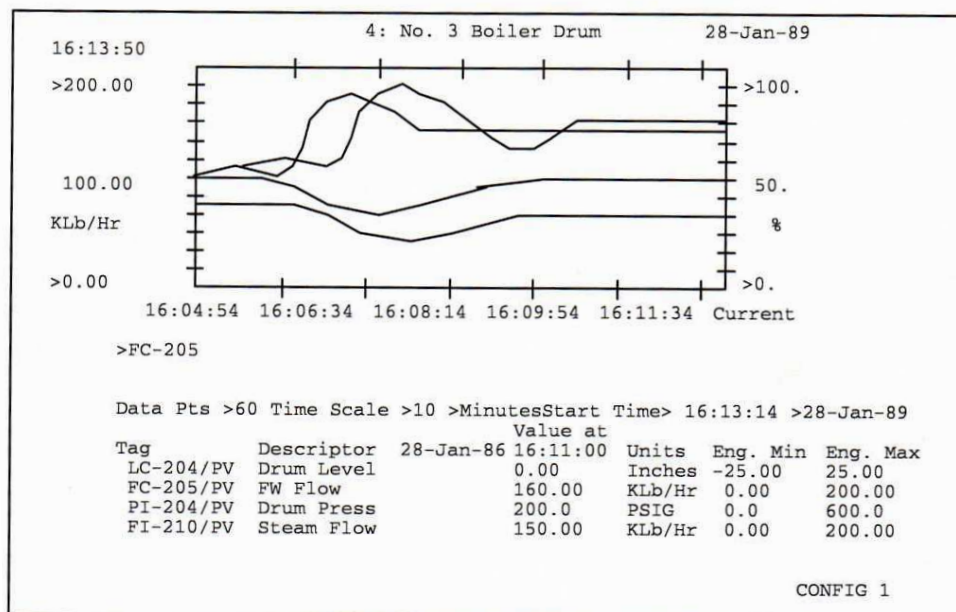


Figura 4.9 - Tendência do RS3 [14].

4.1.5. Interface com Equipamentos Externos

O método desenvolvido pela *Emerson Process Management* para permitir a comunicação fora da rede *PeerWay* baseou-se na utilização de um equipamento denominado por *RS3 Network Interface (RNI)*.

Um *RNI* é um *gateway* que liga o *RS3* com aplicações que correm em outros computadores fora do sistema. O *RNI* é ligado directamente ao *PeerWay* (figura 4.10), sendo portanto um nó na rede proprietária e, é ligado à rede *ethernet* pretendida, permitindo a comunicação de dados processuais, de controlo e mensagens de alarme. A comunicação pode ser feita nos dois sentidos, ou seja, é possível ler e escrever no *RS3* a partir de um *PC*.

Os *RNI* foram fundamentais para a substituição da utilização na operação processual de *MTCC* por *PC*, utilização de *softwares* de controlo avançado e criação de uma base de dados em tempo real de utilização corporativa. No entanto, a utilização das *MTCC* continua a ser imprescindível ao nível da configuração do *RS3*, visto que, os *PC* apenas possibilitam a operação.

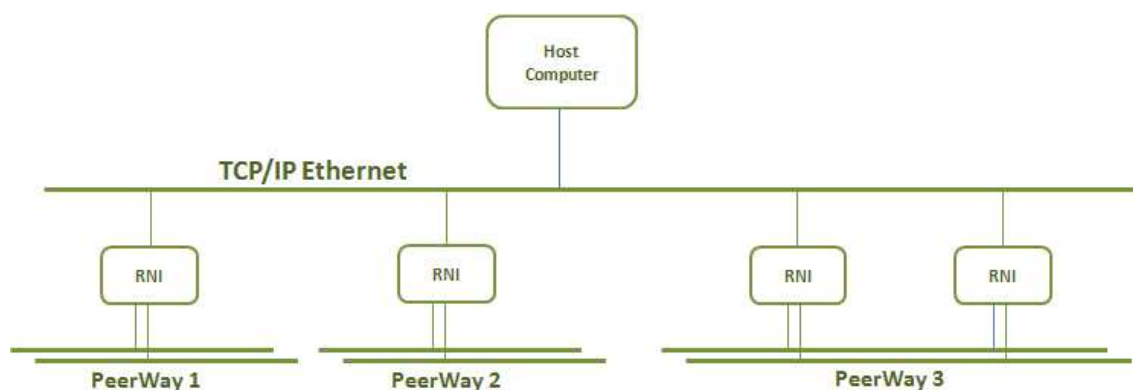


Figura 4.10 - Ligação de *RNI* à rede *ethernet* e ao *PeerWay* [15].

4.1.6. Estrutura e Funcionalidades do *Software*

O *software* do *RS3* é baseado em blocos de função (*function blocks*), com os quais é possível construir a estratégia de controlo pretendida. Estes blocos de função residem nas cartas *Controller Processor*, sendo que cada bloco tem uma função predefinida. Os tipos de blocos existentes são:

- Blocos de entrada,
- Blocos de controlo,
- Blocos de saída.

A descrição detalhada de cada um destes tipos de blocos é disponibilizada no anexo 1.

Na mais simples estratégia de controlo é necessário um bloco de cada tipo de forma a ler um sinal de campo (p.e. figura 4.11) e efectuar o cálculo predefinido para enviá-lo para uma saída (p.e. figura 4.12). No entanto, podem ser utilizados diversos blocos de entrada ligados a vários blocos de controlo para actuarem numa só saída, entre outras configurações possíveis. O *RS3* permite, assim, alguma flexibilidade ao nível *software* com configurações mais ou menos elaboradas em que se podem interligar diversos blocos de controlo.

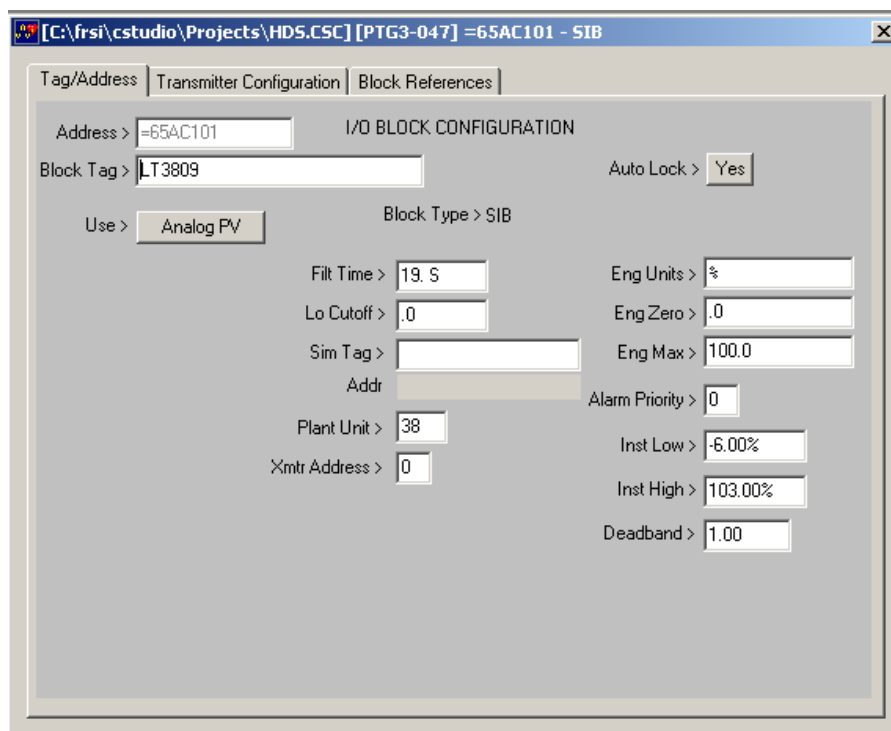


Figura 4.11 - Bloco de entrada analógica *smart*.

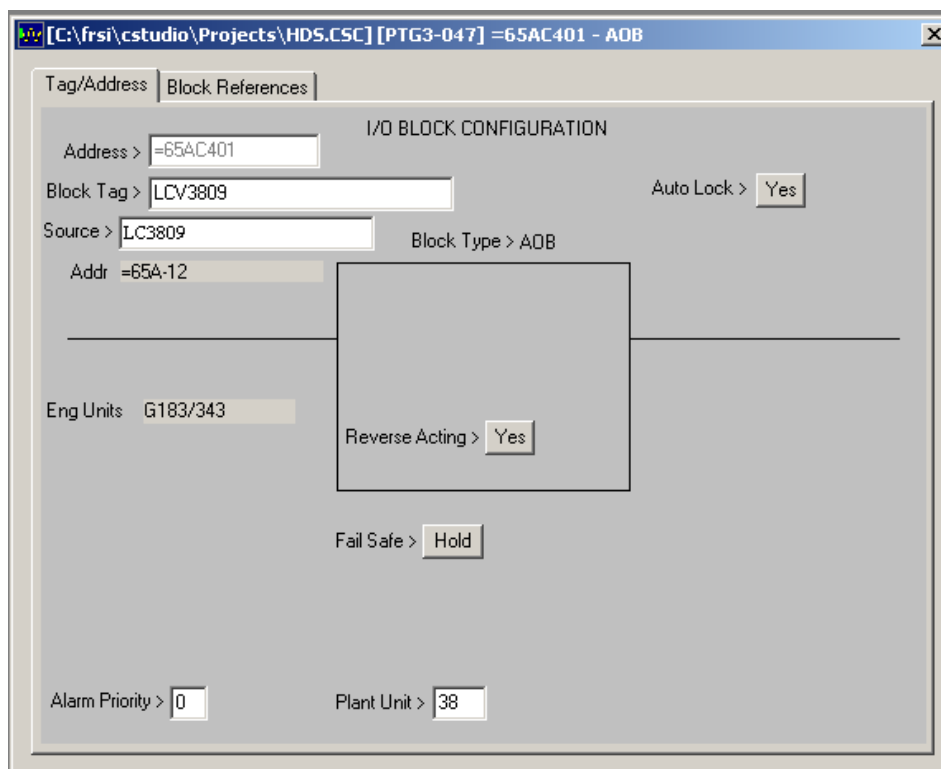


Figura 4.12 - Bloco de saída analógica.

Alterações a efectuar às estratégias de controlo podem ser realizadas *online*, minimizando o impacto no processo.

Nas figuras 4.13 e 4.14 são demonstradas as estratégias de controlo típicas para a configuração de um controlo analógico e um controlo digital, respectivamente. A primeira utiliza como exemplo o controlo de uma pressão e a segunda o controlo de um motor.

Seguindo o percurso do sinal de pressão de 4-20 mA até à actuação da válvula de controlo teremos:

1. Transmissor de pressão (4-20 mA),
2. Carta de entrada analógica - *FIC* (*Field Interface Card*),
3. Bloco de entrada analógico (*software*) - *AIB* (*Analog Input Block*),
4. Bloco de controlo (*software*) - *CB* (*Control Block*),
5. Bloco de saída analógico (*software*) - *AOB* (*Analog Output Block*),
6. Carta de saída analógica - *FIC*,
7. Válvula de controlo recebendo o sinal de 4-20 mA.

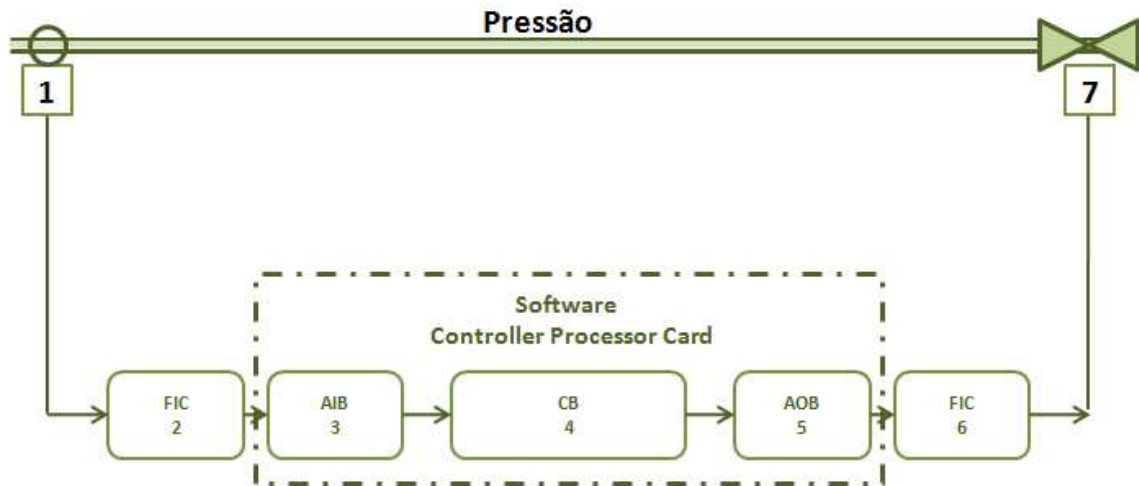


Figura 4.13 - Loop de controlo contínuo típico do RS3.

Em relação ao sinal digital teremos:

1. Sinal de campo *ON/OFF*,
2. Carta de entrada digital - *FIC* ou *FIM* (*Field Interface Card* ou *Field Interface Module*),
3. Bloco de entrada digital (*software*) - *DIB* (*Discrete Input Block*),
4. Bloco de controlo (*software*) - *CB*,
5. Bloco de saída digital (*software*) - *DOB* (*Discrete Output Block*),
6. Carta de saída digital - *FIC* ou *FIM*,
7. Sinal de campo que recebe a saída binária.

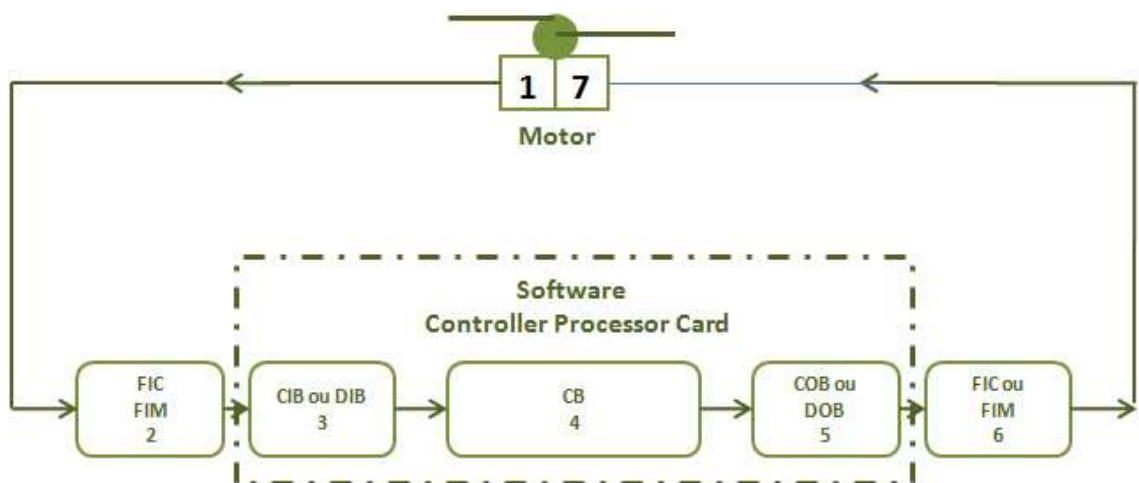


Figura 4.14 - Loop de controlo discreto típico do RS3.

Existem diversos tipos de blocos de controlo no RS3. Alguns deles são específicos para controlar motores, motores de duas velocidades, válvulas motorizadas e arranques em sequência. No entanto, todos os blocos possuem características comuns como, por exemplo, a possibilidade de ligar a cada bloco um conjunto de 15 entradas discretas (de @a a @o), 15 entradas analógicas (de A a O), uma saída analógica (Q) e 16 saídas discretas (passos lógicos de a a p). A figura 4.15 apresenta o diagrama funcional dos blocos de controlo.

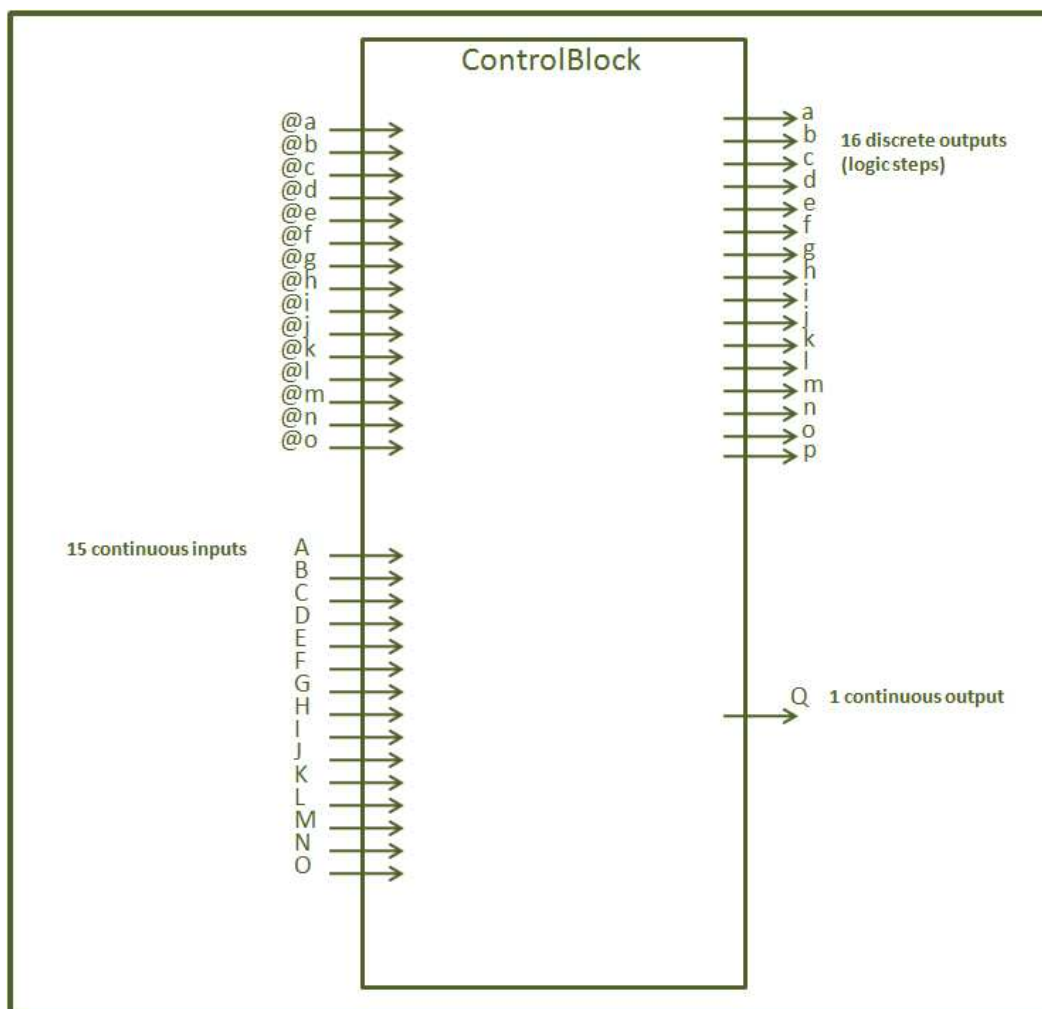


Figura 4.15 - Diagrama funcional dos blocos de controlo [14].

Para permitir que um bloco de controlo se possa ligar a outro são usadas as entradas analógicas e discretas apontando para o bloco de controlo de origem pretendido.

Todos os blocos de saída digitais e analógicos recebem, obrigatoriamente, o comando/valor desde um, e um só, bloco de controlo.

Para variáveis de entrada digital os blocos de controlo aceitam a saída de blocos *DIB* ou *CIB* (*Contact Input Block*), saídas digitais de outros blocos de controlo ou entradas do utilizador do tipo *On*, *Off*, *Toggle* e *Select*.

Nas variáveis de entrada analógicas os blocos de controlo permitem saídas de blocos *AIB*, saídas analógicas de outros blocos de controlo, entradas do utilizador (*Entry*), valores calculados (*Value*), contadores (*Count*) e temporizadores (*Timer*).

Com este conjunto de entradas são efectuadas as lógicas necessárias nos passos lógicos de *a* a *p* do *Discrete Block Diagram* que está dividido em duas partes: condições e acções. As condições que permitem que passo lógico seja activado podem ser de três tipos: definida na linha de “=>” que sempre que for a um o passo é imediatamente activado e se for zero o passo ficará inactivo, o *Set* que permite activar o passo logo que o resultado da condição seja 1 e permanece mesmo que esta transite para zero esperando que o passo *Clear* “limpe” o que foi activado pela função *Set*, figura 4.16.

As acções podem ser de quatro tipos: transição positiva ou negativa e quando o passo está activo ou inactivo. A acção a executar em cada caso poderá passar por efectuar cálculos (figura 4.17), activar outros passos lógicos, determinar saídas analógicas, alterar parâmetros do próprio bloco de controlo como é o caso das acções proporcional, integral e derivativa para o caso dos blocos de controlo do tipo *PID*.

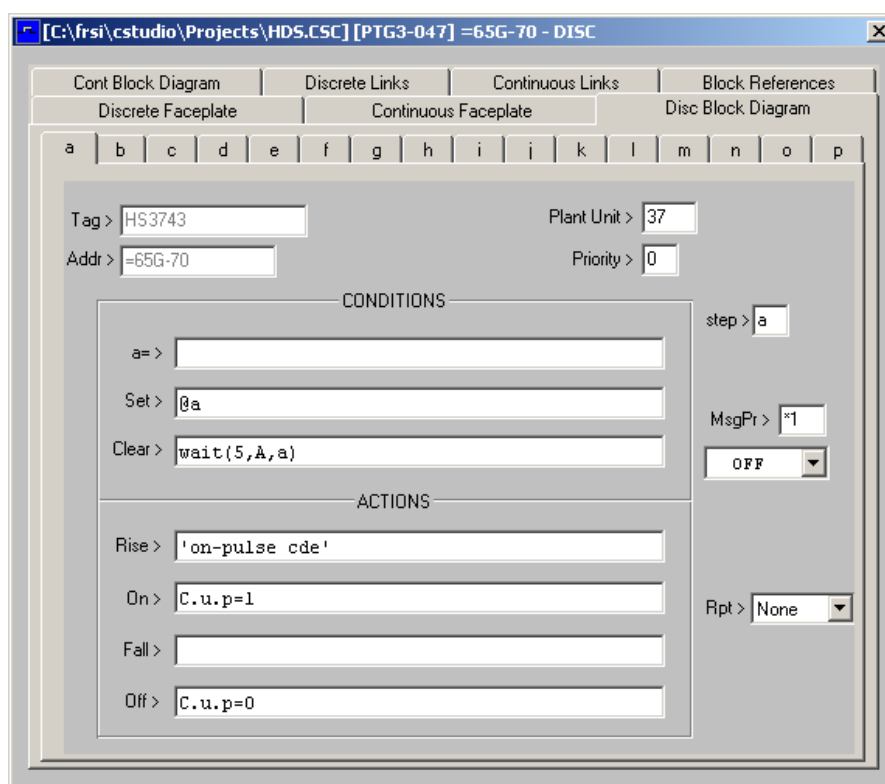


Figura 4.16 - Passo lógico *a* de um bloco de controlo - *Set* e *Clear* e acções em *On* e *Off* com um comentário adicionado no *Rise*.

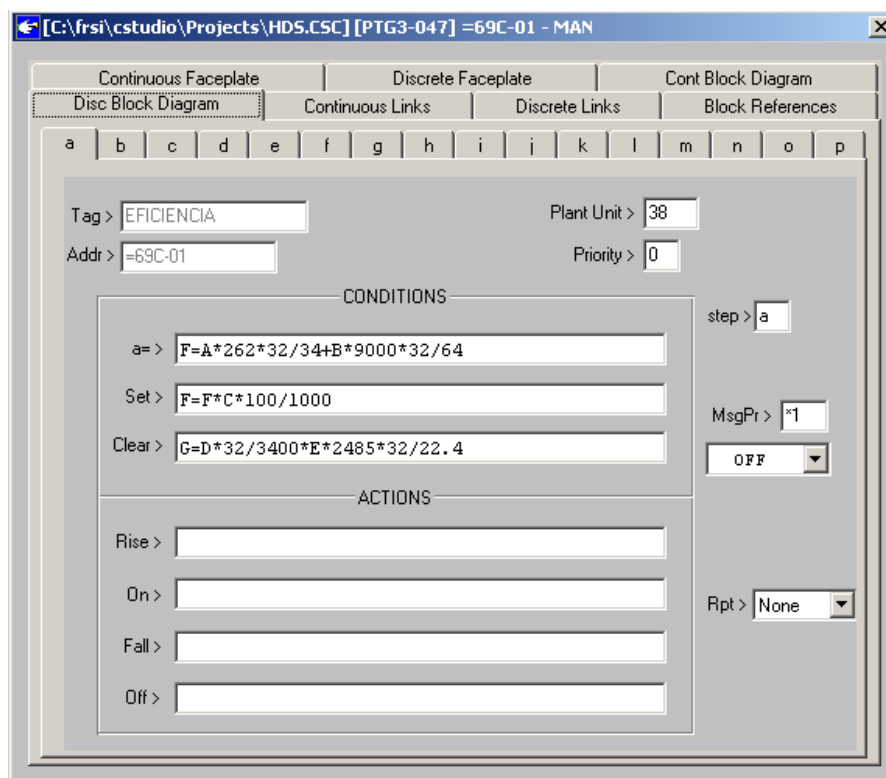


Figura 4.17 - Passo lógico *a* de um bloco de controlo - Cálculos.

4.1.7. Endereçamento

O endereçamento é a convenção usada para garantir que os dados são enviados dos módulos de entrada e saída do campo para o sistema, do sistema para as entradas e saídas, de um bloco de controlo para outro, assim como para as consolas.

Todo o *hardware* que compõe o sistema necessita de um endereço único que o identifique e que possibilite a sua programação no *software*.

Um endereço típico de um bloco de entrada ou saída tem o formato “=65AB105” (figura 4.18), enquanto que o endereço típico de um bloco de controlo é por exemplo “=65A-22” (figura 4.19). A leitura destes endereços, da esquerda para a direita, dá-nos a localização dos dados a que se pretende aceder, com um nível de detalhe crescente, sendo estes precedidos pelo símbolo “=”.

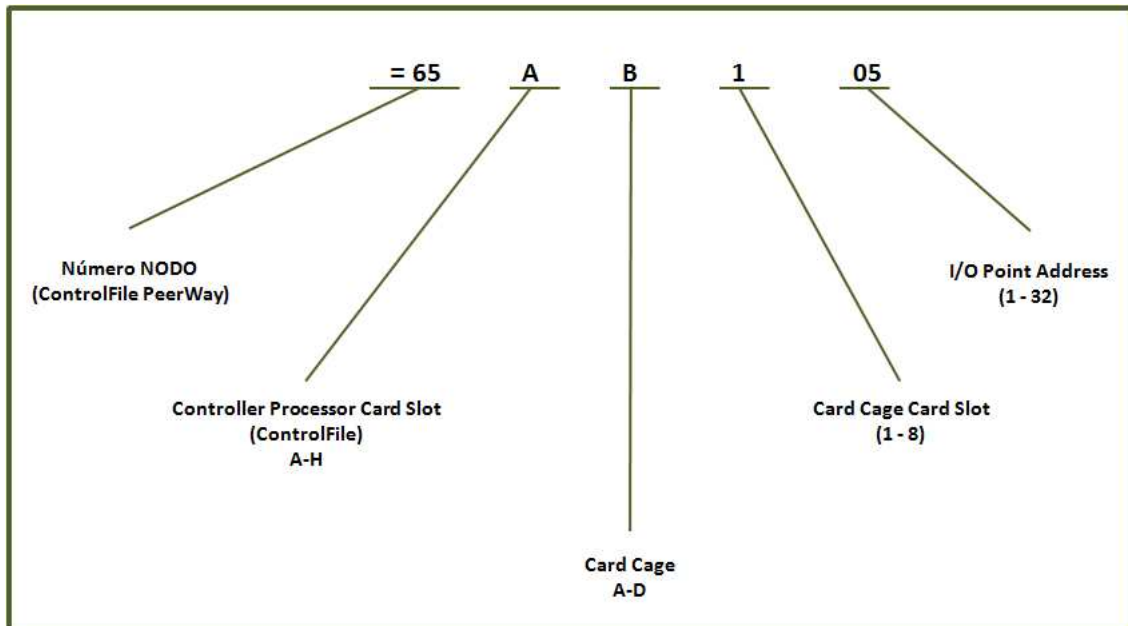


Figura 4.18 - Endereço típico de um bloco de entrada ou saída.

O primeiro elemento de um endereço é o número do nó do *PeerWay*, sendo este definido no *hardware* recorrendo a *jumpers* de configuração. O elemento seguinte diz respeito ao endereço da ranhura em que está localizado o *Controller Processor Card Slot* (de A a H), ao passo que o terceiro elemento define o endereço da *Card Cage* (de A a D). Quanto ao quarto elemento, este indica a *Card Cage Slot* e, em que FIC está inserida (de 1 a 8) e o último elemento corresponde ao endereço do ponto de entrada ou saída composto por dois dígitos de 01 a 32, dependendo do tipo de carta utilizada.

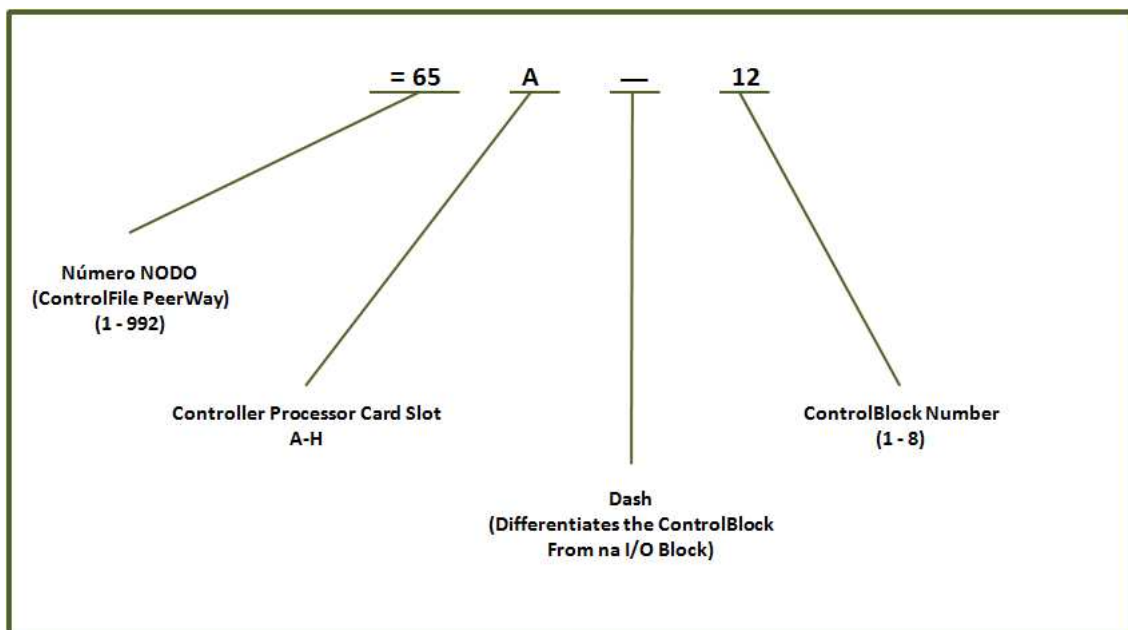


Figura 4.19 - Endereço típico de um bloco de controlo.

No caso do endereçamento de blocos de controlo o primeiro e segundo elementos representam, de igual forma, o nó do *PeerWay* e o *Controller Processor Card Slot*, respectivamente. A grande diferença entre o endereçamento de blocos de entrada e saída reside no terceiro elemento que é um símbolo “-“. Os dois dígitos que se seguem correspondem ao número do bloco de controlo (de 1 a 265 para as *MPC IV*).

4.1.8. RS3 na refinaria de Matosinhos

Na refinaria de Matosinhos existem cinco barramentos de comunicação interligados entre si através de *HIA*, conforme a figura 4.20.

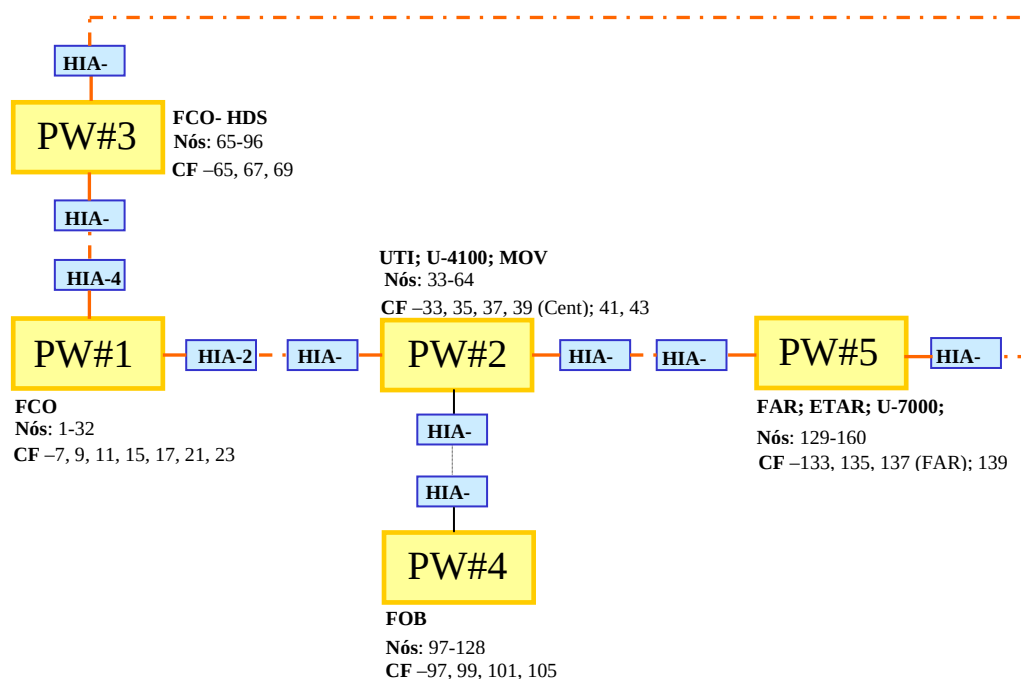


Figura 4.20 - Arquitectura do RS3 na refinaria de Matosinhos.

Os cinco barramentos estão relacionados com as fábricas existentes, com excepção do *PeerWay* 3 que corresponde a uma unidade específica da fábrica de combustíveis - *HDS*, e do *PeerWay* 2 que compreende duas fábricas *FUT* e *MOV*.

A arquitectura do sistema é mista, formando um anel fechado com quatro dos barramentos, permitindo que, em caso de perda de comunicação, esta seja efectuada no sentido oposto. O *Peerway* 4 encontra-se ligado ao anel através do *PeerWay* 2, sendo o único a ficar isolado no caso de uma perda de comunicação. As perdas de comunicação referidas permitem, contudo, que o controlo local (dentro do próprio *PeerWay*) funcione normalmente, apenas afectando o controlo que seja dependente de variáveis oriundas de outros barramentos.

As consolas de operação em ambiente *Windows* estão ligadas através do *RNI* ao *PeerWay*. A figura 4.21 mostra essa ligação para a fábrica de utilidades, bem como as *ControlFiles* e *HIA* do *Peerway 2*.

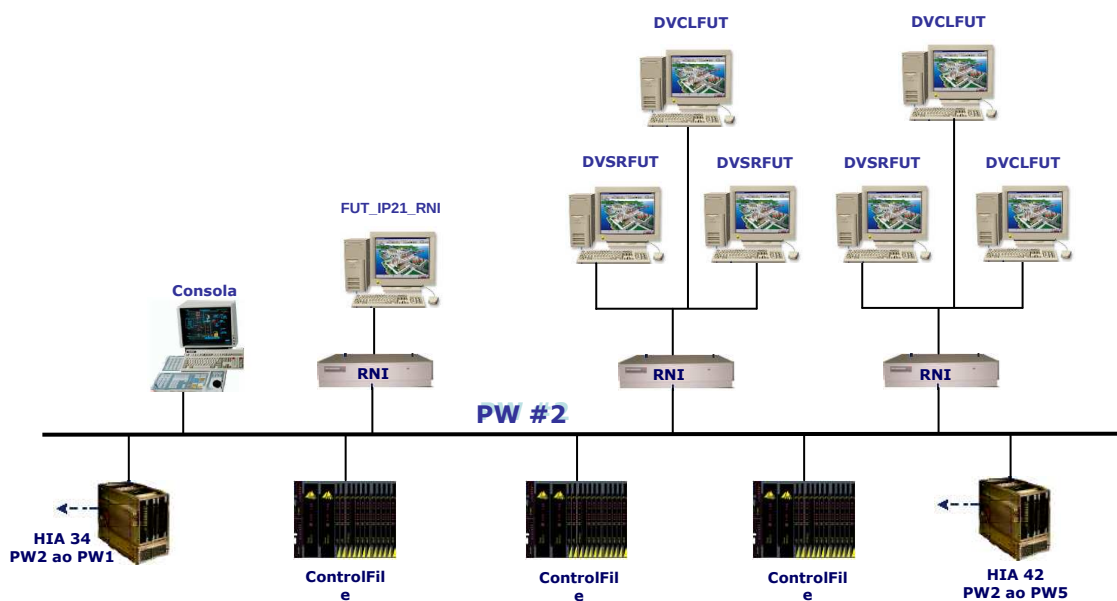


Figura 4.21 - Nós do *PeerWay 2*.

4.2. DeltaV

O *DeltaV* é o sistema de controlo distribuído mais recente da *Emerson Process Management* e conta já com 10 versões sendo que, na refinaria de Matosinhos, se encontra instalada a versão 9.3.1.

Ao nível de *hardware* o *DeltaV* é composto por (figura 4.22):

- Estações de trabalho,
- Barramento de comunicação que pode ser, ou não, redundante,
- Fontes de alimentação,
- Pelo menos um controlador que pode ser ou não redundante,
- Subsistema de *E/S* e,
- Identificador do sistema (chave física única que contém a licença que permite identificar o sistema).

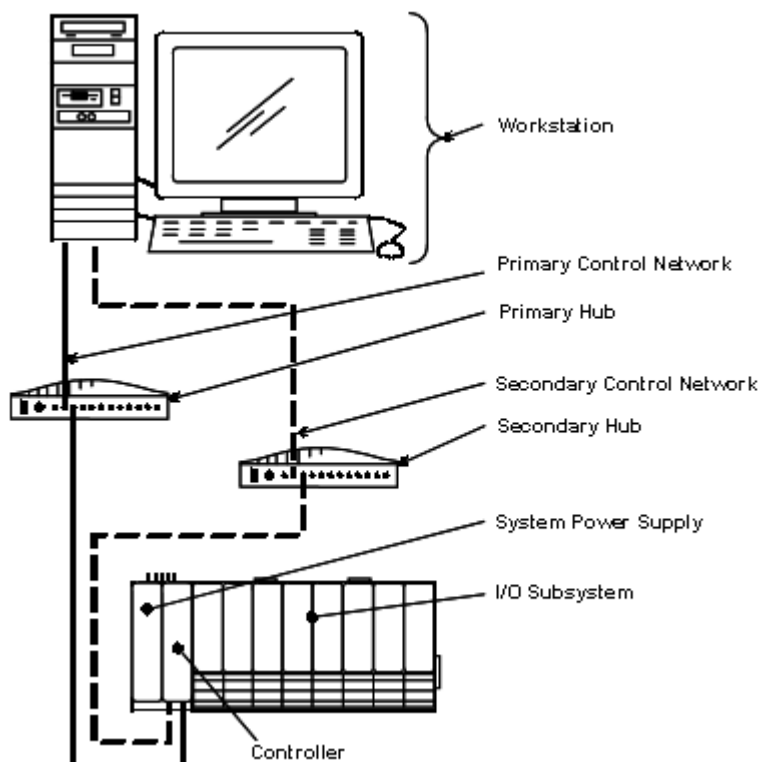


Figura 4.22 - Arquitectura de *hardware* básica do *DeltaV* [16].

Em relação à capacidade do sistema, são permitidos no máximo 120 nós distribuídos da seguinte forma:

- Máximo de 100 controladores,
 - Máximo de 60 estações de trabalho,
- e são permitidos até 30 000 sinais de E/S.

4.2.1. Barramento de comunicação

A comunicação no *DeltaV* é baseada na tecnologia *Ethernet* sendo, portanto, aberto à comunicação com outros sistemas de controlo, de monitorização, de optimização, entre outros. Permite, igualmente, a utilização da tecnologia como *OPC*, *FOUNDATION™ Fieldbus*, *DeviceNet* e *Profibus*, tornando-o flexível e versátil.

Para aumentar a robustez do sistema de controlo é usada uma rede de comunicação *Ethernet* redundante tanto para comunicação de dados entre controladores, bem como destes para as estações de trabalho. Este barramento de comunicação é exclusivo para o *DeltaV* não sendo permitida a ligação de outros sistemas com pena de perda de suporte por parte da *Emerson Process Management*.

Na figura 4.23 apresenta-se a rede de comunicação com quatro controladores e quatro estações de trabalho sendo, portanto, constituída por oito nós, usando dois *switches* ou *hubs*, um para a rede primária e outro para a rede secundária. Quando a distância entre nós e *switches/hubs* é superior a 100 m são utilizados cabos de fibra óptica de forma a interligar os *switches/hubs* de comunicação, para garantir a fiabilidade do sinal.

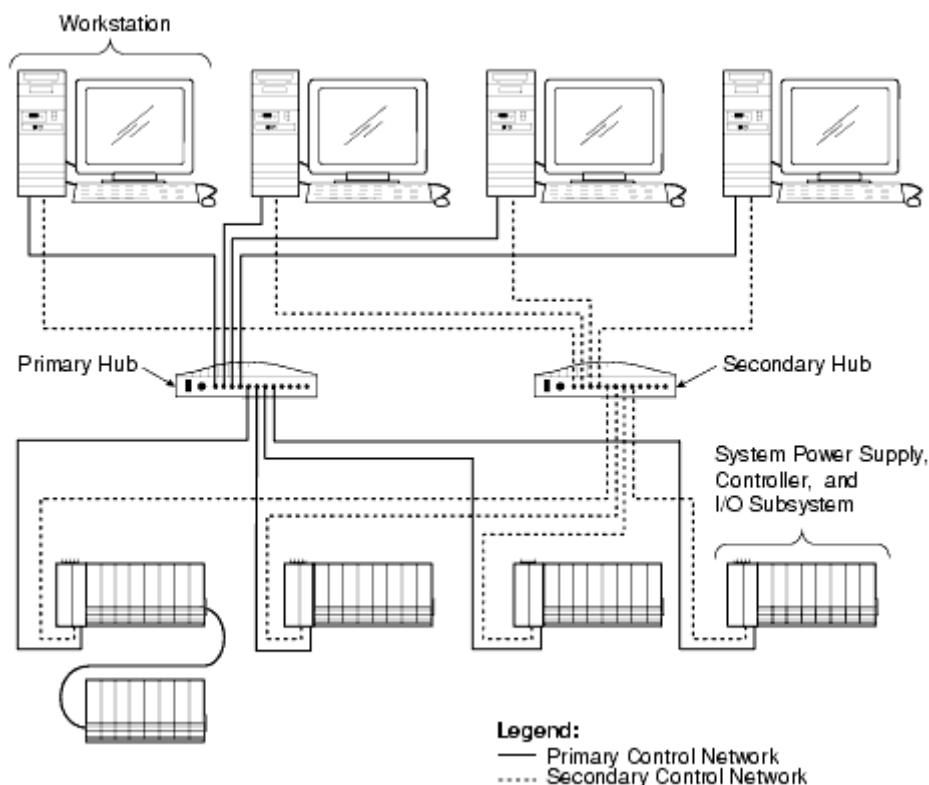


Figura 4.23 - Rede de comunicação com oito nós: quatro controladores e quatro estações de trabalho [16].

4.2.2. Controladores

Os controladores são responsáveis pelo controlo local e gerem as comunicações entre o subsistema de *E/S* e o barramento de comunicação. Todos os algoritmos de controlo são processados no controlador, podendo gerir um máximo de 64 cartas *E/S* por controlador simples ou redundante e processar 750 *E/S* com tempos de execução mínimos de 100 ms.

Os controladores são montados em bastidores de quatro posições (2 controladores e duas fontes de alimentação) fixados em calha *DIN*, conjuntamente com uma ou duas fontes de alimentação (se o controlador for simples ou redundante) - figura 4.24.

Estas fontes de alimentação disponibilizam 12 ou 24 V *DC* para alimentação da electrónica das cartas, contudo não alimentam os sinais *E/S* e estão limitadas a um máximo de 8 A de consumo.

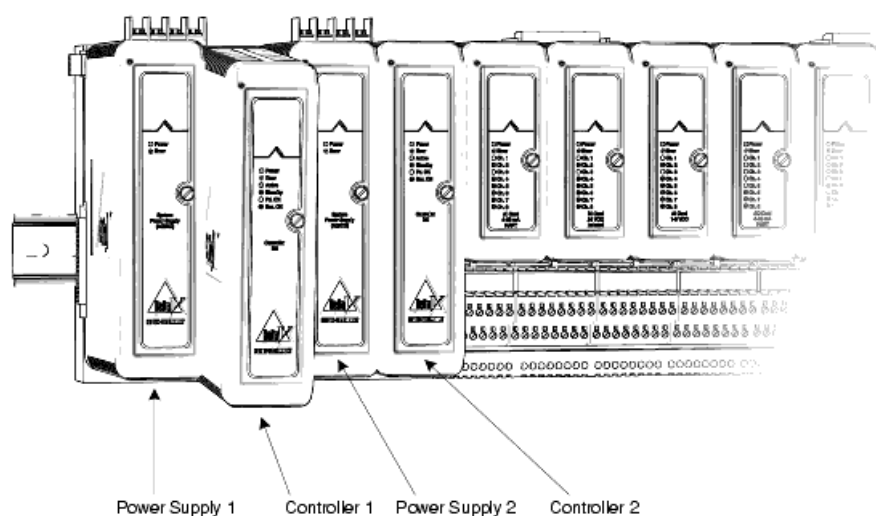


Figura 4.24 - Controladores e fontes de alimentação do sistema redundantes montados em calha DIN [16].

No caso de ser excedido o consumo de 8 A por parte das cartas E/S a configuração da alimentação dos *carriers* terá que ser alterado conforme a figura 4.25. A fonte de alimentação passará a alimentar *carriers* de E/S posteriores aos alimentados pelas fontes de alimentação adjacentes aos controladores.

É, assim, necessário efectuar um cálculo detalhado do consumo do sistema para que não ocorra sub-dimensionamento que poderá provocar paragens intempestivas de unidades processuais.

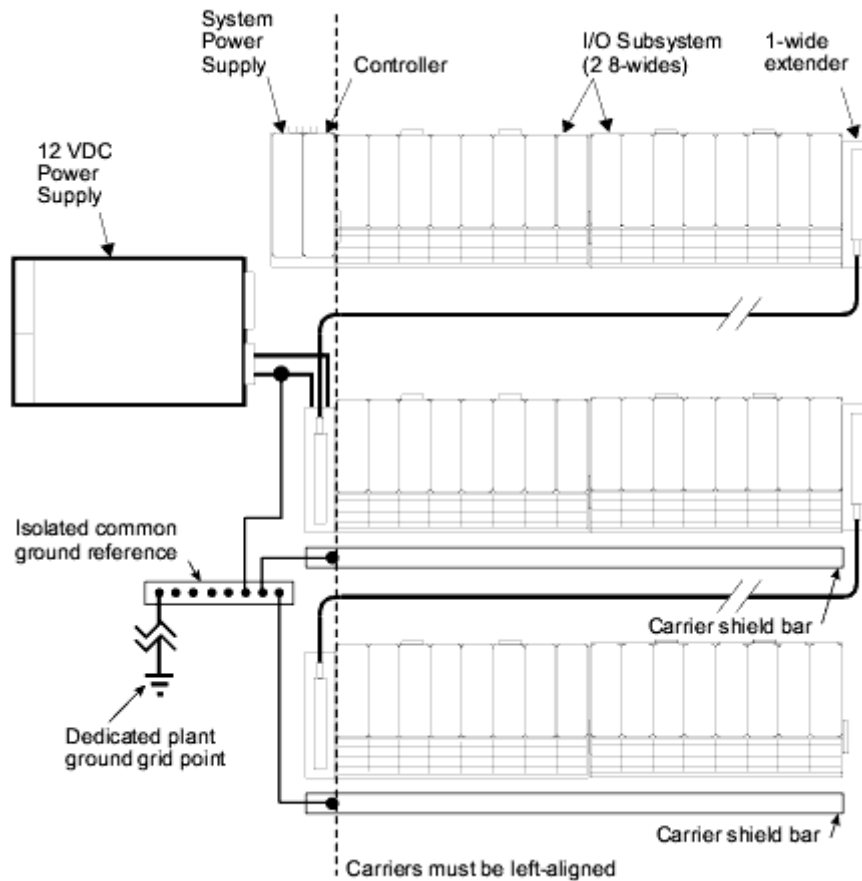


Figura 4.25 - Alimentação de *carriers* a partir das fontes de alimentação do sistema e a partir da fonte de alimentação externa [16].

4.2.3. Cartas de E/S

O *DeltaV* disponibiliza diversos tipos de cartas de entrada e saída conforme o tipo de sinal ou consoante a aplicação pretendida. As cartas possuem pontos de ligação denominados canais e são montadas em *carriers* de oito posições, com o auxílio dos *terminal blocks* (figura 4.26).

As cartas de utilização mais frequente são:

- Cartas de entradas analógicas (8 ou 16 canais),
- Cartas de saídas analógicas,
- Cartas de entradas digitais (8 ou 32 canais),
- Cartas de saídas discretas (8 ou 32 canais),
- Carta H1 para *FIELDBUS™ foundation*,
- Carta série de 2 portas (comunicação *MODBUS*) e,
- Carta para termopares em mV.

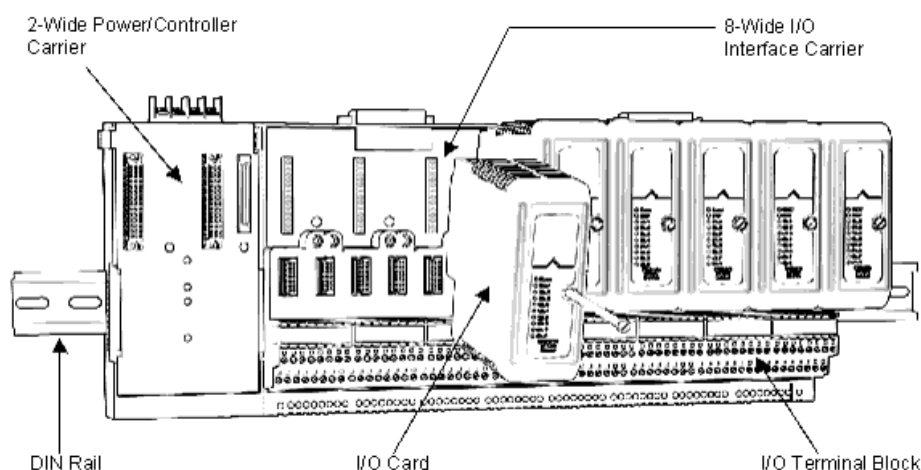


Figura 4.26 - *Terminal blocks* e cartas *E/S* montadas num *carrier* de oito posições [16].

A alimentação dos sensores e actuadores pode ser feita a partir do *DeltaV*, desde fontes de alimentação de 24 V DC (figura 4.27). A alteração desta alimentação implica a alteração do tipo de *terminal block* a utilizar, sendo que a carta *E/S* se mantém a mesma.

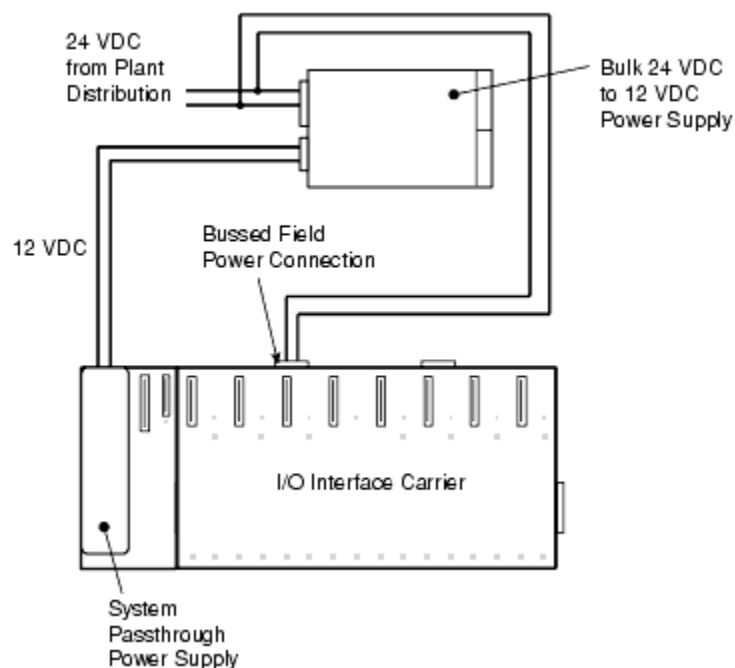


Figura 4.27 - Alimentação dos dispositivos de campo a partir da fonte de alimentação externa de 24 V DC [16].

4.2.4. Estações de trabalho

As estações de trabalho permitem a configuração, operação e manutenção de todo o *software* do *DeltaV*, incluindo diagnósticos aos dispositivos de *hardware*. O tipo de estações de trabalho depende não só do tipo de instalação de *software* mas também do tipo de licença adquirida.

Existem quatro tipos de estações de trabalho:

- *ProfessionalPLUS* (apenas 1 no sistema),
- Estações de engenharia (10 no máximo),
- Estações de operação, manutenção e *base* (59 no máximo) e,
- Estações *Application* (10 no máximo).

O número total de estações não poderá, contudo, ultrapassar o máximo estabelecido no sistema, ou seja, 60.

Na *ProfessionalPLUS* é guardada toda a base de dados que contém toda a informação de configuração do sistema, incluindo controladores, cartas *E/S* e estações de trabalho. É nesta estação que se liga a chave física identificadora do sistema.

Quando se acede a uma configuração de um sinal por parte de outras estações é sempre à *ProfessionalPLUS* que estará a ser efectuado o acesso e, portanto, só poderá existir uma estação deste tipo no sistema. Este acesso será exclusivo, ou seja, mais nenhuma outra estação poderá aceder naquele momento ao mesmo módulo.

As estações de engenharia permitem fazer todo o tipo de configuração tal como a *ProfessionalPLUS*, no entanto, não é nestas estações que reside a base de dados. Estes dois primeiros tipos de estações de trabalho permitem realizar alterações *offline* de forma a não afectar o processo mas também disponibilizam um método para alterações *online*.

As estações de operação, manutenção e *base* diferem no tipo de licença instalada. A primeira possibilita a operação processual utilizando a aplicação *DeltaV Operate*. A segunda possui *software* de manutenção *Asset Management Software* (AMS) que possibilita o diagnóstico, configuração e calibração de instrumentação e equipamentos de campo, e a estação *base* possui apenas a instalação base, em que as restantes aplicações poderão ser adicionadas posteriormente.

A *Application Station* guarda o histórico de variáveis processuais previamente configuradas na base de dados, com uma taxa de compressão que pode ser definida pelo configurador. As restantes estações de trabalho podem aceder à base de dados histórica simultaneamente. A comunicação usando a tecnologia *OPC* é baseada na utilização desta estação, sendo esta a interface para outras aplicações externas como bases de dados ou *softwares* de utilização corporativa.

As estações de trabalho são *PC* com sistema operativo *Windows Server* (*ProfessionalPLUS* e por vezes nas *Application Stations*), 2000 ou *Windows XP* (restantes estações de trabalho).

4.2.5. Estrutura e Funcionalidades do *Software*

A organização da base de dados do *DeltaV* é feita de forma hierárquica, sendo o primeiro nível denominado por área. Dentro de uma área é possível definir directamente módulos, mas é também possível definir células processuais, unidades, ou módulos de equipamento onde, por sua vez, residem os módulos.

Um módulo tem um nome único e é constituído por ligações a sinais *E/S* ou a parâmetros de outros módulos, pelo algoritmo de controlo, alarmes, condições, *displays* e definição dos parâmetros a serem guardados em base de dados de históricos (figura 4.28).

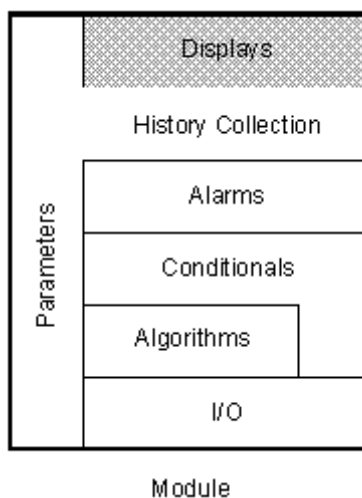
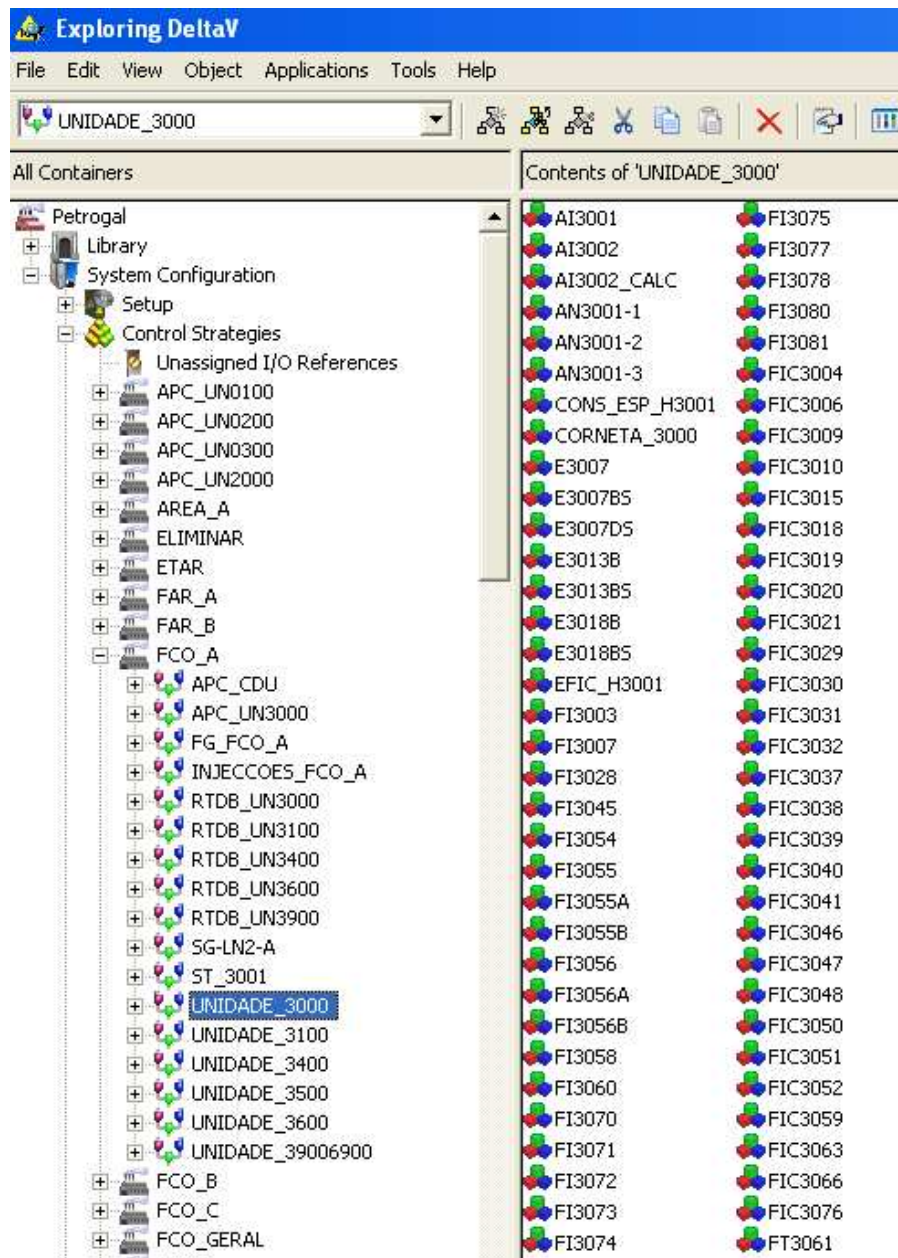


Figura 4.28 - Componentes básicos de um módulo de *DeltaV* [16].

A utilização de células processuais, unidades, ou módulos de equipamento permite uma organização do sistema semelhante ao do processo em si, facilitando a procura de módulos e a sua organização. Os módulos de equipamento permitem ainda efectuar uma gestão de alarmes específica, onde são agrupados os alarmes referentes a um dado equipamento.

A estrutura do sistema pode ser consultada no *DeltaV Explorer* que é semelhante ao *Windows Explorer*, com a configuração e expansão em árvore desde o topo da hierarquia até ao elemento mais básico (figura 4.29).

Figura 4.29 - Hierarquia do *DeltaV*.

O *DeltaV* possui uma biblioteca com módulos pré-definidos para implementar as mais usuais estratégias de controlo. A biblioteca pode ser populada pelo configurador com novos blocos ou poderá mesmo alterar os existentes.

Na biblioteca existem também os blocos de função que o configurador não poderá alterar (um exemplo na figura 4.30). São constituídos por diversos parâmetros que o configurador pode ocultar ou mostrar, ligando o parâmetro pretendido através de um conector gráfico até ao parâmetro do bloco de função seguinte. O configurador poderá desenvolver blocos de função próprios que se adaptem à sua unidade processual. Todos os algoritmos existentes ou a desenvolver são definidos recorrendo à linguagem *Function Block Diagram* ou *Sequential*

Function Chart especificados na norma IEC61131-3-*Programmable Controllers-Programming Languages* [17].

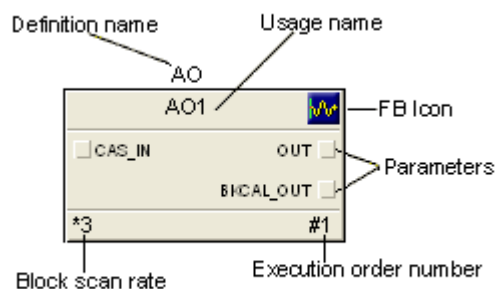


Figura 4.30 - Bloco de função de saída digital com três parâmetros visíveis [16].

A figura 4.31 mostra a utilização de blocos de função interligados para criar a estratégia de controlo residente num módulo de *DeltaV*. Estes blocos de função obedecem ao *standard IEC61804 - Function Block for process control* [18-19]. No anexo 2 é apresentada breve descrição dos blocos de função do *DeltaV* para a versão 9.3.1.

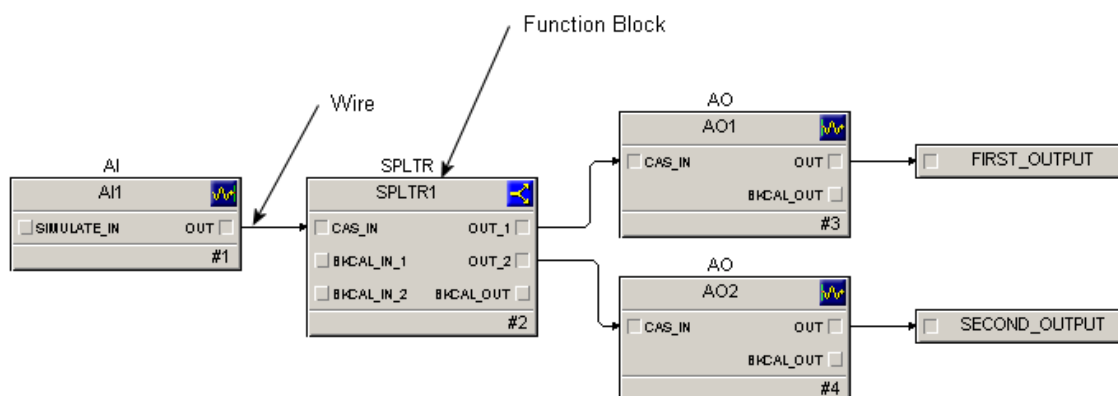


Figura 4.31 - Estratégia de controlo simples onde uma entrada analógica actua em duas saídas analógicas.

O configurador poderá ainda definir classes de módulos de controlo que têm que ser adicionados obrigatoriamente à biblioteca de classes. A utilização de uma classe implica criação de instância, ou seja, cada vez que essa classe é usada é criada uma instância que está sempre ligada à classe. Quando um parâmetro é alterado na classe, ou seja, na biblioteca, esse parâmetro será actualizado em todas as instâncias. A utilização de classes deve ser bem estruturada visto que a alteração poderá acontecer quando as instâncias já se encontram em funcionamento.

Contrariamente, a alteração de qualquer outro módulo na biblioteca não implica a alteração dos que foram criados a partir dele, nem a alteração de um módulo já em funcionamento implica a alteração do módulo na biblioteca. Não é mantida qualquer ligação ao módulo original.

Ao nível do *software* as diversas aplicações que o *DeltaV* pode facultar ao utilizador/configurador são apresentadas no anexo 3, sendo divididas, fundamentalmente, em dois grupos: ferramentas de engenharia e ferramentas de operação.

As ferramentas de engenharia permitem, configurar módulos e estratégias de controlo, gestão das contas de utilizadores, administração da base de dados e gestão de dados históricos entre outras potencialidades. As ferramentas de operação permitem monitorizar o processo, avaliar o desempenho e fazer diagnósticos simples de instrumentação/equipamento de campo.

4.2.6. Endereçamento

No *DeltaV* o endereçamento é utilizado apenas para os sinais de *E/S* e registos provenientes da comunicação *MODBUS* ou outro protocolo suportado pelo *DeltaV*.

Para que um módulo de *DeltaV* seja capaz de receber o sinal *E/S* de campo necessita de um endereço que terá a seguinte forma: Nome_Controlador/Número_Carta/Número_Canal. Este endereço é único para o sinal e também é atribuído um nome ao canal da carta *E/S*, que tem que ser igualmente único no sistema.

Caso o sinal corresponda a um registo *MODBUS* o endereçamento será diferente e da forma:

Nome_Controlador/Número_Carta/Número_Porta/Número_Device/Número_Dataset/Número_Registo ou, de forma mais simples, Nome_Tag_Dataset/Número_Registo. A utilização de uma ou outra nomenclatura prende-se exclusivamente com o gosto pessoal do configurador.

Qualquer outro módulo no sistema não possui endereçamento, sendo referenciado sempre pelo nome do módulo.

4.2.7. *DeltaV* na refinaria de Matosinhos

Uma rede de fibra óptica interliga todos os *switches* da rede primária e secundária de *DeltaV* da refinaria de Matosinhos. A *star*, localizada na FCO, é um *switch* que gere as comunicações entre todas as fábricas do complexo, sendo assim possível a comunicação de dados entre todos os controladores e estações de trabalho, existindo uma *star* por cada uma das redes redundantes.

A velocidade de comunicação é de 100 *Mbit*, estando já pensada a alteração da rede para 1 *Gbit*.

A figura 4.32 mostra a arquitectura de hardware da rede primária do *DeltaV*.

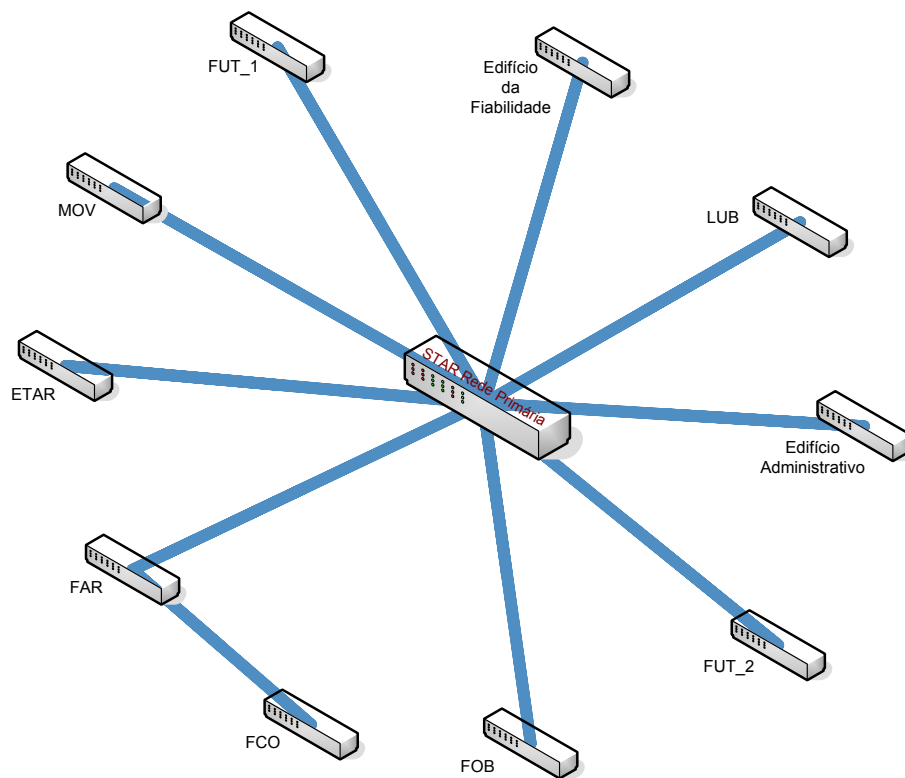


Figura 4.32 - Arquitectura da rede primária do *DeltaV* da refinaria.

Capítulo 5

Discussão do Problema

Com este capítulo pretende-se descrever o método utilizado actualmente na migração ao nível de software desde o *RS3* para o *DeltaV*, sem preocupação de especificar a solução proposta, visto ser alvo de capítulo próprio.

Serão também explicadas, com detalhe, as ferramentas utilizadas na configuração *online* e *offline*, importação/exportação e migração das configurações entre os dois sistemas. No caso da análise de módulos de gestão de válvulas e bombas será explorada a ferramenta de configuração da interface de operação.

5.1. Ambiente de Desenvolvimento do *RS3*

O *RS3* tem um ambiente de desenvolvimento *online* caracterizado por ser pouco amigável, relativamente simples ao nível das configurações de *E/S* e muito complexo na configuração das lógicas de controlo. Na consola *MTCC* (consola de operação e configuração do *RS3* - secção 4.1.4) é possível efectuar configurações em linha com o processo, sendo necessário tomar medidas preventivas de forma a minimizar o impacto processual, como por exemplo colocar a malha de controlo no modo manual.

De forma a descrever o processo de configuração no *RS3*, será dado o exemplo de configuração de uma entrada analógica do tipo *AIB* no endereço =139AD104. Através da consola *MTCC* é introduzido o endereço pretendido e verifica-se que em *Block Type* aparecerá *NONE*, conforme a figura 5.1. De realçar, na mesma figura, que no campo *Device Type* surgem os tipos de blocos que são permitidos para aquele endereço: *AIB* e *SIB*, sendo que qualquer outra entrada será considerada inválida pelo sistema. No presente caso deveria ser seleccionado o tipo de bloco *AIB*. A partir desta selecção surgirão no ecrã as opções possíveis para serem introduzidas manualmente (figura 5.2), como por exemplo, o *tag*, descritivo, filtro de tempo, alarmes, ou unidades.

Address =139AD104		I/O BLOCK CONFIGURATION	21-Aug-09 08:32:20
		Device Type AIB or SIB	
		AIB	
		Block Type >NONE	
Block Not Configured			

Figura 5.1 - Configuração de uma entrada analógica AIB no RS3.

Address =139AD104		I/O BLOCK CONFIGURATION	21-Aug-09 08:33:41
		Device Type AIB or SIB	
		AIB*	
Block Tag >		Block Type >AIB	Mode > AUTO
Field Value -25.00			Auto Lock >no
-25.00 %			Block Out .00
----->		Filt Time >None	----->
Eng Units >		Sig Char None	
Eng Zero >.00		Lo Cutoff .00	Nom Out >None
Eng Max >100.00			Inst Bias .00
Descriptor >		----- ALARMS -----	
Inst High >None		Crit High >None	Adv High >None
Inst Low >None		Crit Low >None	Adv Low >None
Hardware Alarm Code 0	Priority >0	Plant Unit >0	Al DdBand >1.00

Figura 5.2 - Configuração de uma entrada analógica AIB no RS3 e valores colocados por defeito logo após a criação do bloco.

As novas configurações, bem como as alterações em blocos existentes, ficam automaticamente activas, não sendo necessário efectuar qualquer *download*/carregamento de configuração. Este facto pode ser perigoso, pois qualquer erro fica activo após premir a tecla *return/enter*.

Um erro frequente consiste na eliminação de blocos após premir *return/enter* em cima do *Block Type*, visto que elimina imediatamente o bloco sem solicitar qualquer confirmação ao utilizador, tendo que ser criado novamente.

Outro erro igualmente frequente consiste em eliminar o tag, sem eliminar a sua configuração. Este caso acontece quando é premida a tecla *return/enter* em *Block Tag* que elimina o nome do bloco.

Em ambos os casos a operação perde o acesso ao valor do campo e a todos os outros valores dependentes deste (como por exemplo cálculos) visto ter sido eliminado o tag.

De forma a minimizar este e outros erros que eventualmente possam ser cometidos pelo configurador, existe uma ferramenta para consulta e configuração *offline* dos blocos lógicos: o *Control Studio (CS) for RS3*. Esta ferramenta permite importar e exportar configurações para/de um PC, através de um RNI (interface para *ethernet* - secção 4.1.5), directamente para/da consola MTCC. Assim sendo, a base de dados *online* pode ser guardada num formato próprio (.csc) que permite a sua visualização, consulta e alteração num PC sem afectar o processo.

A utilização do *CS for RS3* é fundamental para a consulta de todos os sinais de *E/S* existentes no sistema de controlo, bem como para o estudo das estratégias de controlo implementadas, de forma a replicar essa mesma configuração no *DeltaV*, sem se ter que estar ligado directamente ao sistema de controlo *RS3*, evitando afectar o processo.

A configuração da lógica de controlo utilizando blocos de controlo é efectuada recorrendo ao *Discrete Block Diagram*, após a correcta configuração de todos os *links* contínuos e discretos necessários.

5.2. Ambiente de Desenvolvimento do *DeltaV*

O ambiente de desenvolvimento do *DeltaV* é bastante simples e amigável para o utilizador.

De forma a descrever o processo de configuração no *DeltaV* será dado como exemplo a configuração de uma entrada analógica no controlador *CTRL-FUT-2*, carta 5, canal 1 (figura 5.3). Após a selecção das propriedades do canal que se pretende configurar, basta introduzir o tag desejado em *Device Tag*, e colocar um visto em *enable*, para que o canal se torne activo. Em seguida é necessário criar o módulo de leitura do valor analógico proveniente da referida carta/canal. Para tal recorre-se ao *DeltaV Explorer*. Conforme a figura 5.4, basta seleccionar *New > Control Module* e voltar a introduzir o tag pretendido.

Abrindo o módulo com a ferramenta de edição/configuração *Control Studio* do *DeltaV* (figura 5.5) é possível configurar os restantes parâmetros como descritivo, escala, alarmes, entre outros.

Quando se efectua uma configuração esta não fica automaticamente activa como no *DCS RS3*. Para que a configuração fique activa é sempre necessário efectuar o *download* das cartas e módulos novos ou afectados por eventuais alterações. O *download* permite colocar as novas configurações (de *hardware* - cartas, canais, portas ou de *software* - módulos) no controlador de *DeltaV* responsável pela execução dos mesmos.

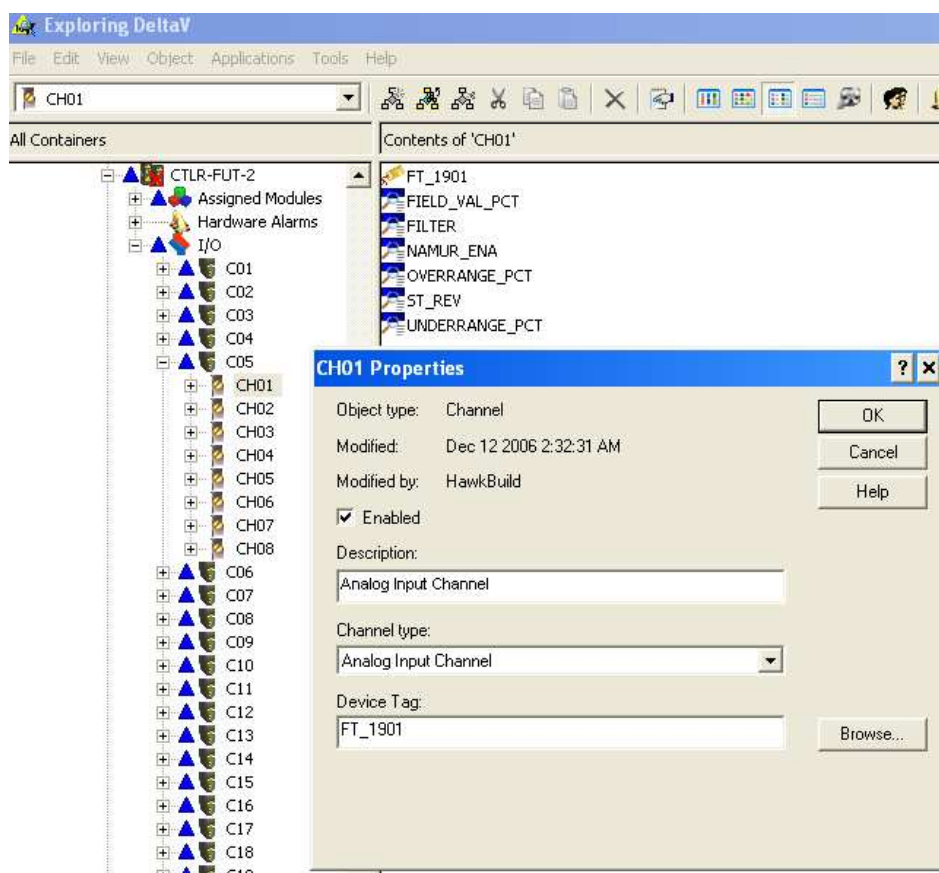


Figura 5.3 - Configuração de um canal para entrada analógica na carta 5 do controlador *CTLR-FUT-2* do *DeltaV*.

O *DeltaV* possui uma ferramenta semelhante à do *RS3*, que permite a consulta *online* ou *offline* da configuração implementada. No caso do *DeltaV* não é possível exportar para *PC* os blocos da mesma forma que o *CS for RS3* mas possibilita a importação/exportação para uma folha de cálculo *MS Excel* após criação de um ficheiro de especificação de formato (ferramenta *bulk edit* detalhada na secção seguinte), ou a importação/exportação segundo um formato próprio (.FHX), característico do *DeltaV*. Este formato pode ser analisado recorrendo a ferramentas de texto, no entanto e, de forma a ser compreendido pelo sistema, é necessário utilizar aquela extensão.

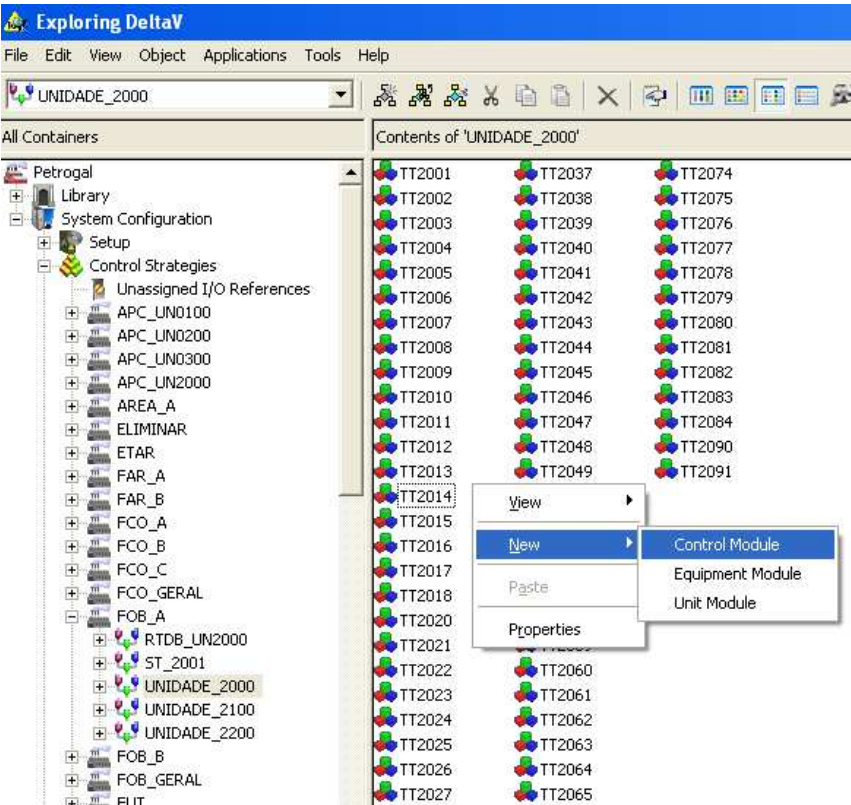


Figura 5.4 - Criação de um novo módulo no *DeltaV*.

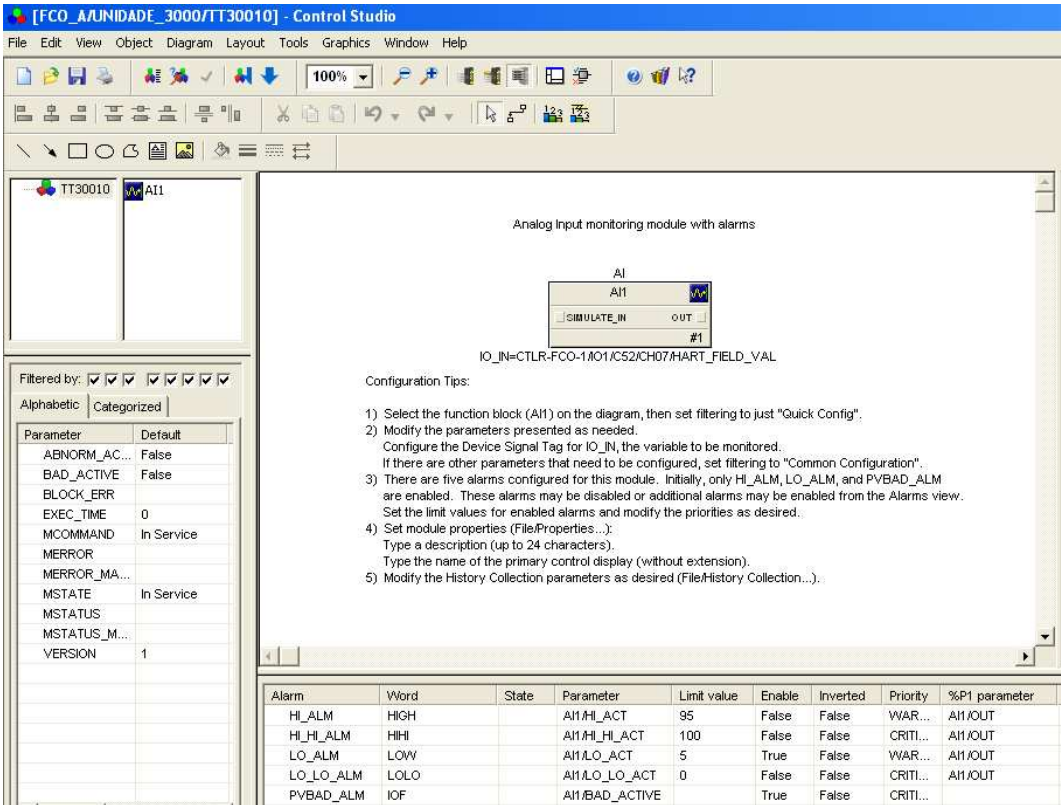


Figura 5.5 - *Control Studio* do *DeltaV*: configuração de uma entrada analógica (AI1) a partir de um transmissor inteligente (HART_FIELD_VAL).

5.3. Processo de Migração do RS3 para o DeltaV

O primeiro sinal a ser mudado para *DeltaV* foi configurado manualmente, consultando a configuração existente no *CS for RS3* e replicando-a no *CS do DeltaV*. Este processo é moroso visto que implica a anotação manual de dados como *tag*, descritivo, escalas, unidades, etc., sendo que, para cada bloco analógico ou digital sem lógica adicional, poderá demorar cerca de 10 minutos a configurar. Após as primeiras migrações de sinais do *RS3* para *DeltaV* (em 2005), tornou-se cada vez mais premente a utilização de ferramentas automáticas de migração de forma a minimizar o tempo dispendido na sua configuração.

Com o amadurecimento e formação de engenheiros foi possível iniciar a configuração de módulos típicos. Estes módulos são o mais generalista possível, de forma a corresponderem à maior parte dos blocos existentes no *RS3*. Os módulos típicos residem na biblioteca do *DeltaV Explorer* (figura 5.6) e podem ser alterados conforme necessário, afectando apenas novos módulos criados a partir da alteração. Para malhas de controlo, malhas de controlo em cascata, válvulas e bombas foram, assim desenvolvidos módulos típicos e *faceplates* adequados à sua correcta operação.

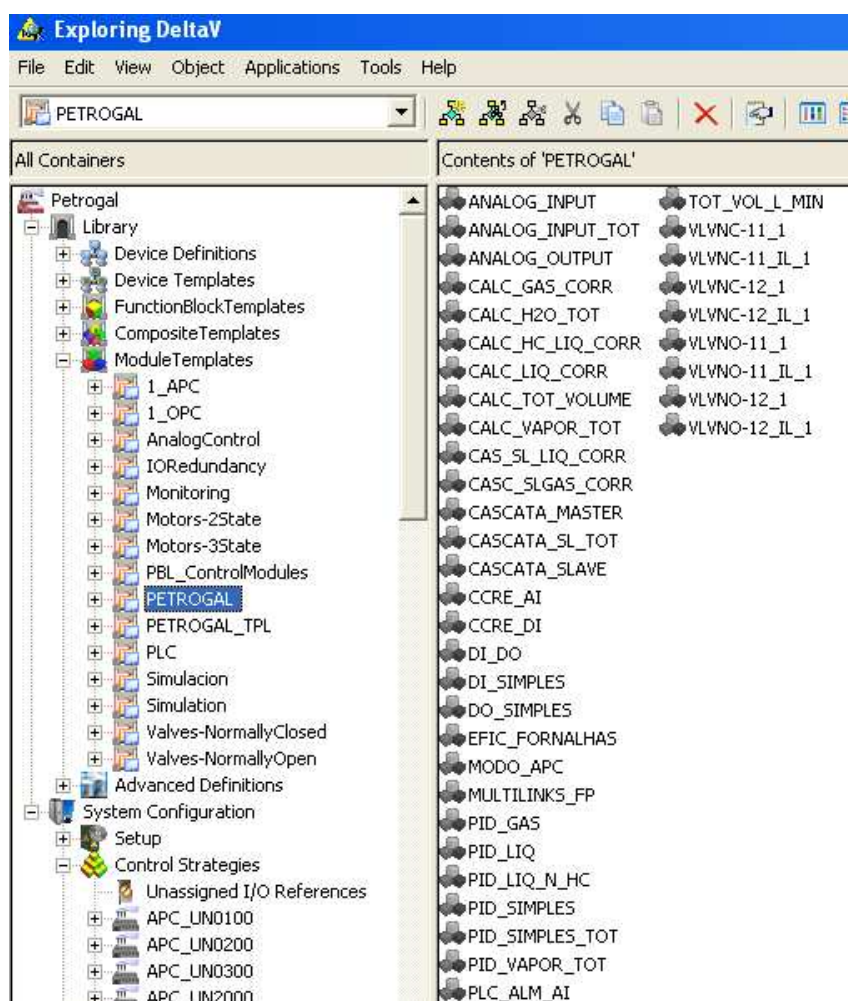


Figura 5.6 - Biblioteca com os módulos típicos criados no *DeltaV* para a refinaria.

Em Agosto de 2005 foi utilizada pela primeira vez na refinaria a ferramenta do *DeltaV* denominada *Bulk Edit*. Esta ferramenta permite a configuração dos sinais no *DeltaV* utilizando a folha de cálculo *MS Excel* apoiada por ficheiros de especificação de formato pré definidos pelo utilizador ou já instalados aquando da instalação do *DeltaV* (figura 5.7).

Esta ferramenta não permite a migração automática entre sistemas, apenas permite minimizar o tempo de criação dos módulos, ou seja, se são necessários criar 200 módulos semelhantes em que muda o endereço físico, descritivo, *tag* e escala, usa-se esta ferramenta em que apenas se terá que criar um deles manualmente e todos os outros serão criados à imagem do anterior com as devidas alterações que serão inseridas na folha de cálculo (figura 5.8).

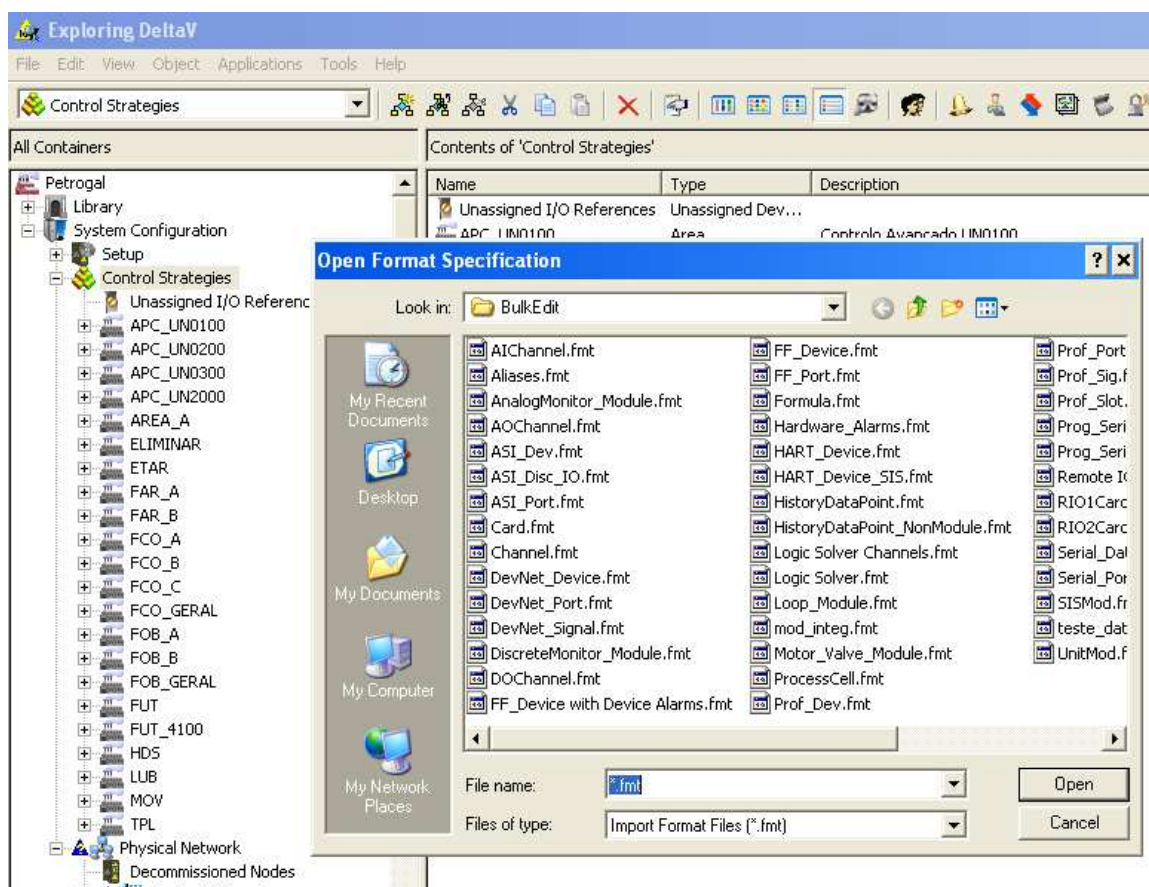


Figura 5.7 - Ficheiros de especificação de formato existentes no *DeltaV*.

Para criar um módulo no *DeltaV* utilizando esta ferramenta é necessário conhecer previamente o formato a cumprir. Assim sendo, deve ser criado um formato de base utilizando o *DeltaV Explorer*. Uma vez criado o formato poderemos aceder aos campos a preencher através do *MS Excel*. A extensão deste ficheiro é *FMT* que, mais uma vez, pode ser editado usando uma ferramenta de texto (figura 5.9).

	A	B	C	D	E	F
1	module_name	description	ctrlr	area	work_in_progress	restore_param_vals
2	TT0129	Entrada R-0151A/B	CTLR-FAR-1	FAR_A/UNIDADE_100	F	F
3	TT0128	Saida E-0102	CTLR-FAR-1	FAR_A/UNIDADE_100	F	F
4	TT0124	Saida E-0111	CTLR-FAR-1	FAR_A/UNIDADE_100	F	F
5	TT0123	Topo T-0102	CTLR-FAR-1	FAR_A/UNIDADE_100	F	F
6	TT0110	Alimentacao U-100	CTLR-FAR-1	FAR_A/UNIDADE_100	F	F
7	TT0109	Topo T-0101	CTLR-FAR-1	FAR_A/UNIDADE_100	F	F
8	TT0108	Prato 40 T-0101	CTLR-FAR-1	FAR_A/UNIDADE_100	F	F
9	TT0107	Prato 25 T-0101	CTLR-FAR-1	FAR_A/UNIDADE_100	F	F
10	TT0106	Prato 12 T-0101	CTLR-FAR-1	FAR_A/UNIDADE_100	F	F
11	TT0105	Prato 5 T-0101	CTLR-FAR-1	FAR_A/UNIDADE_100	F	F
12	TT0104	Fundo T-0101	CTLR-FAR-1	FAR_A/UNIDADE_100	F	F
13	TT0103	Alimentacao T-0101	CTLR-FAR-1	FAR_A/UNIDADE_100	F	F
14	TT0102	C8 Saida E-0101	CTLR-FAR-1	FAR_A/UNIDADE_100	F	F
15	TT0101	C8 SaidaE-0104	CTLR-FAR-1	FAR_A/UNIDADE_100	F	F
16						

Figura 5.8 - Exportação de módulos segundo o formato especificado na fig. 5.2.

[illegible]

Figura 5.9 - Conteúdo de um ficheiro de especificação de formato.

Em alguns casos não é possível a criação de formatos base por diversos motivos, entre os quais: alguns parâmetros são ocultos e, portanto, não poderão ser utilizados quando se cria o ficheiro de especificação de formato e a grande variabilidade do número de parâmetros dentro do módulo que conduz a uma variabilidade do número de colunas a ser preenchida na folha de cálculo. Nestes casos o caminho a seguir é utilizar a importação/exportação *standard* do *DeltaV* (figura 5.10). A extensão destes ficheiros é FHX.

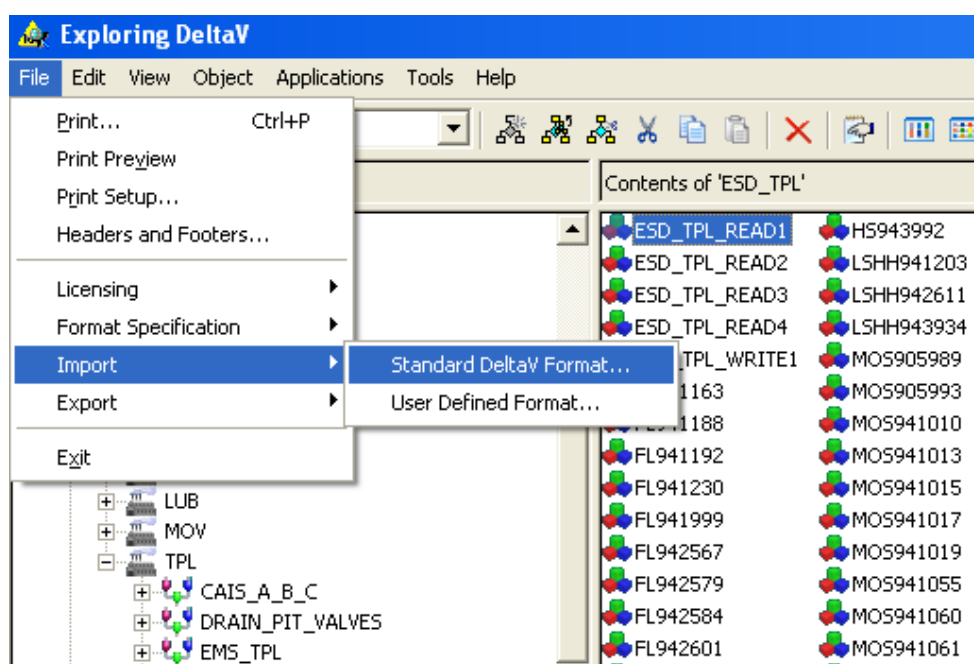


Figura 5.10 - Importação usando o formato de especificação *standard* do *DeltaV*.

Um ficheiro de configuração *FHX* contém toda a informação do módulo, equipamento, área, carta, etc., dependendo do que é exportado e, assim, neste ficheiro poderão ser alterados, acrescentados ou eliminados parâmetros, blocos de função, descritivos ou até mesmo módulos, obedecendo sempre a um código específico. À semelhança dos anteriores, este ficheiro também pode ser editado recorrendo a ferramentas de texto (figura 5.11).



```

ANALOG_INPU_1.fhx - Notepad
File Edit Format View Help

/* Version: 9.3.1.5246.xr */
/* "28-Mar-2009 20:00:19" */

SCHEMA
user="ADMINISTRATOR" time=1238268739/* "28-Mar-2009 19:32:19" */
{
  VERSION=1212632218/* "05-Jun-2008 03:16:58" */
  MAJOR_VERSION=9
  MINOR_VERSION=3
  MAINTENANCE_VERSION=1
  BUILD_VERSION=5246
  BUILD_ID="xr"
  VERSION_STR="9.3.1.5246.xr"
  ONLINE_UPGRADE=F
}
LOCALE
user="ADMINISTRATOR" time=1238268739/* "28-Mar-2009 19:32:19" */
{
  LOCALE="English_United States.1252"
}
PLANT_AREA NAME="ELIMINAR" INDEX=10
user="ADMINISTRATOR" time=1238170532/* "27-Mar-2009 16:15:32" */
{
  DESCRIPTION="Área Testes"
}
FUNCTION_BLOCK_TEMPLATE NAME="AI"
user="hawkbuild" time=1099977248/* "09-Nov-2004 05:14:08" */
{
  DESCRIPTION="Analog Input"
}
MODULE TAG="ANALOG_INPU_1" PLANT_AREA="ELIMINAR" CATEGORY=""
user="ADMINISTRATOR" time=1235057866/* "19-Feb-2009 15:37:46" */
{
  DESCRIPTION="Control Module"
  PERIOD=1
  CONTROLLER=""
  PRIMARY_CONTROL_DISPLAY=""
  INSTRUMENT_AREA_DISPLAY="AI_FP"
  DETAIL_DISPLAY="AI_DT"
  TYPE="Monitor"
  SUB_TYPE="ANALOG_INPUT"
  ASSIGN_BLOCKS_TO_H1_CARD=F
  FUNCTION_BLOCK NAME="AI1" DEFINITION="AI"
  {
    DESCRIPTION="Analog Input"
    ID=47100006
    RECTANGLE= { X=247 Y=93 H=38 W=139 }
    ATTRIBUTE_USAGE NAME="L_TYPE"
    {
      CATEGORY { CATEGORY=COMMON }
      CATEGORY { CATEGORY=CFGVIEW }
      CATEGORY { CATEGORY=INVALID_ON_REMOTEIO }
    }
  }
}

```

Figura 5.11 - Importação usando o formato de especificação *standard* do *DeltaV*.

Por exemplo, as duas primeiras linhas do ficheiro contêm a versão do *DeltaV* utilizada e a data em que foi efectuada a exportação:

```

/* Version: 9.3.1.5246.xr */
/* "28-Mar-2009 20:00:19" */

```

Para iniciar partes do código dependentes do módulo em si, ou de configurações próprias do *Windows*, são usadas chavetas:

```

ATTRIBUTE NAME="HI_HI_ALM" TYPE=EVENT
{
  CONNECTION=INTERNAL_SOURCE
  RECTANGLE= { X=-50 Y=-50 H=1 W=1 }
}

```

A correcta interpretação deste ficheiro poderá permitir efectuar alterações rapidamente, exportando o ficheiro, alterá-lo conforme pretendido e voltar a importar o ficheiro já alterado. Esta técnica é utilizada quando a alteração é exaustiva e repetitiva em diversos módulos. Obviamente que são necessários cuidados adicionais na verificação deste ficheiro

antes da importação de módulos alterados segundo esta técnica visto que, não existe forma de voltar atrás nem é perguntado ao utilizador se quer ou não substituir um módulo existente, a substituição é sempre feita.

5.3.1. Processo de Migração de Configurações de PLC

Um dos objectivos desta dissertação é conceber uma ferramenta para executar a migração o mais automatizada possível dos blocos de entrada do tipo *PLCB* - *Programmable Logic Controller Block* (figura 5.12). Estes blocos são utilizados para efectuar a interface com um PLC utilizando o protocolo *MODBUS*. Este bloco permite definir um conjunto de parâmetros necessários à comunicação como o número do PLC, a *word address*, e o tipo de dado a ser enviado ou recebido. Estes blocos podem ser de leitura ou de escrita dependendo do parâmetro *Read*: se *Read=normal* então o bloco é apenas de leitura se *Read=no* então será exclusivamente de escrita no PLC.

Visto que a carta do RS3 que permite a comunicação com o PLC tem duas portas (A e B) é necessário definir a porta a ser usada. As interfaces RS232, 422 ou 485 utilizadas pelo protocolo *MODBUS* são definidas através de *jumpers* na própria carta PLC de comunicação, ou seja, é definido fisicamente. Os restantes parâmetros necessários como a paridade, *stop bit*, entre outros, são configurados em páginas específicas no *software*, usando a *MTCC*.

The screenshot shows the 'I/O BLOCK CONFIGURATION' dialog box. The 'Tag/Address' tab is selected. The 'Address' field is set to '=33D096'. The 'Block Tag' field is set to 'SGD-SG16'. The 'Source' field is empty. The 'Block Type' is set to 'PLCB'. The 'Data Type' is set to 'Flags'. The 'Sample Time' is set to '1 S'. The 'Sim Tag' field is empty. The 'Port' is set to 'B' and the 'PLC number' is set to '176'. The 'Read' field is set to 'normal'. The 'PLC Word Address' is set to '5098'. Below these fields is a table of flags (a through p) with corresponding addresses and descriptions.

Flags	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p
a:	CFRB-D	e:	SPCP-D	i:	RSRA-D	m:	S2-D									
b:	COP-D	f:	ST40201H-D	j:	LAH40311-D	n:	FCV40201-D									
c:	SPAC-D	g:	ST40201HH-D	k:	LAL40314-D	o:	FIM_DE_PURGA-D									
d:	SPC-D	h:	RSND-D	l:	LOP-D	p:	L4-C_D									

At the bottom of the dialog, the 'Priority' is set to '0' and the 'Plant Unit' is set to '120'.

Figura 5.12 - Bloco de entrada de *flags* de comunicação *Modbus*.

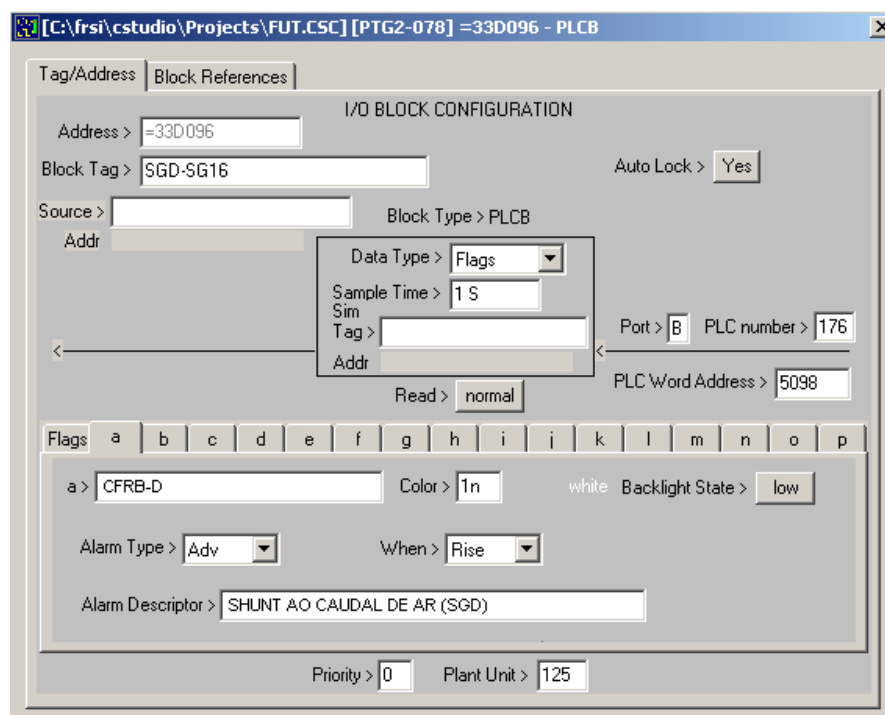


Figura 5.13 - Bloco de entrada de *flags* de comunicação *Modbus*: seleccionado o *bit a* com o respectivo alarme e descritivo associado.

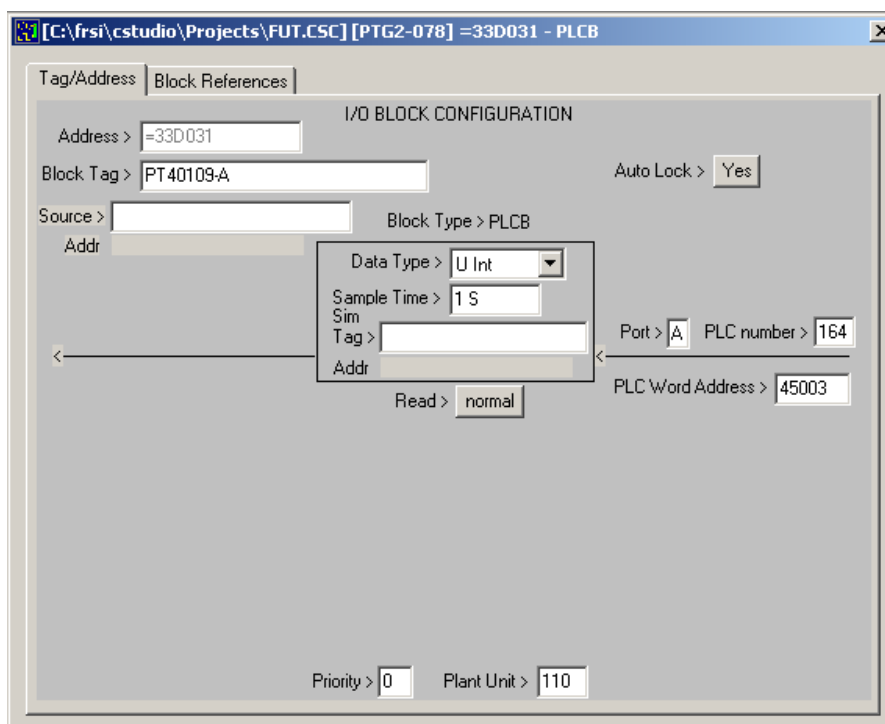


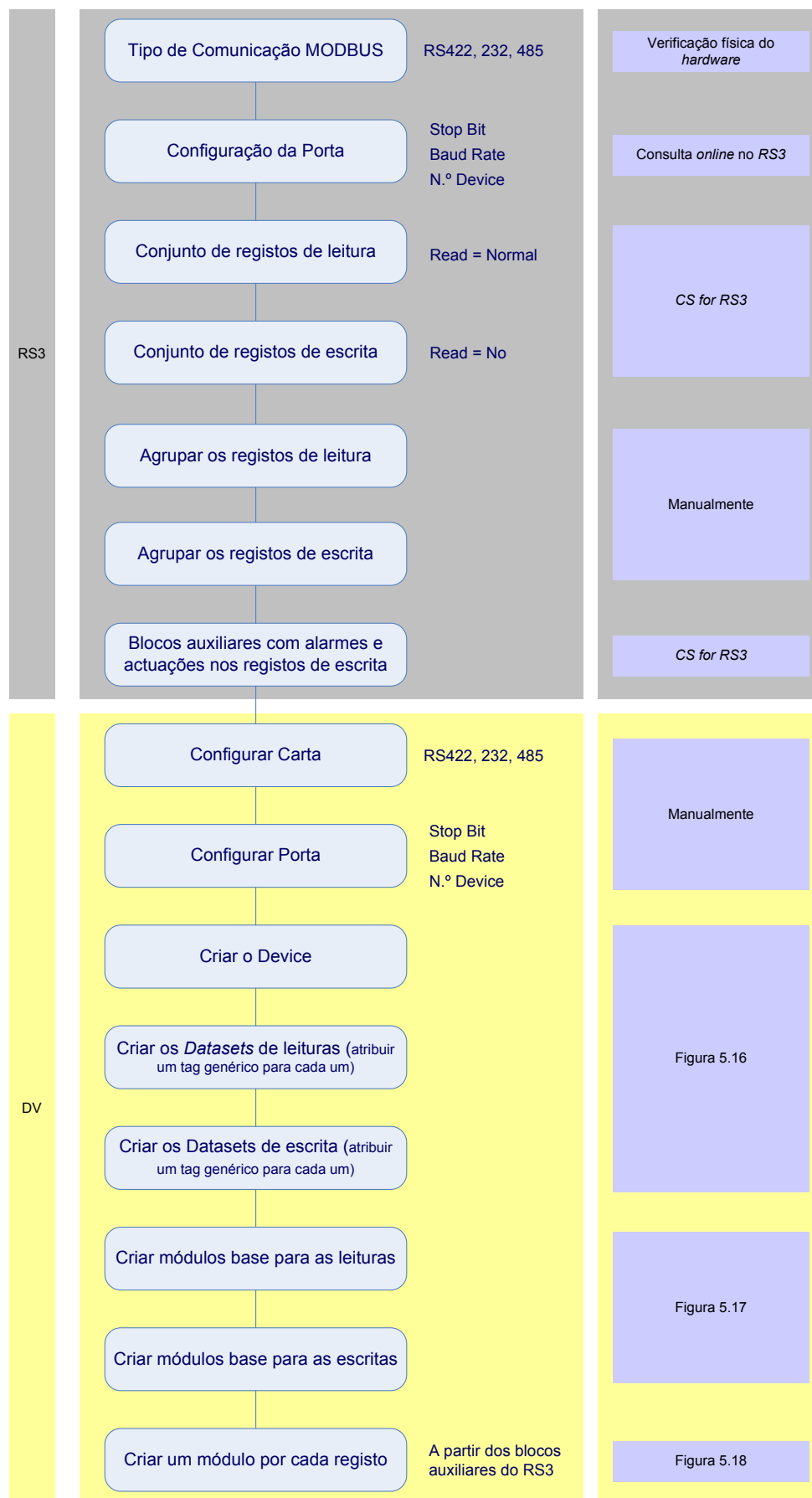
Figura 5.14 - Bloco de entrada de *U Int* de comunicação *Modbus*.

A migração das configurações de comunicação *MODBUS* com *PLC* do *RS3* para o *DeltaV* é, actualmente, efectuada manualmente, consultando todos os blocos do *CS for RS3* (figuras 5.13 e 5.14) relacionados com o *PLC* pretendido.

Alguns elementos de configuração têm que ser consultados directamente no *RS3*, não estando disponíveis na ferramenta *offline*, como o tipo de comunicação, *baud rate*, paridade e *stop bits*. Este aspecto obriga a uma recolha cuidadosa de todos os parâmetros de comunicação bem como de todos os bits, endereços, *tags* e alarmes de forma a não originar módulos com configuração deficiente no *DeltaV*.

Visto que na refinaria a comunicação *MODBUS* é usualmente utilizada na transferência de valores provenientes de *PLC* de segurança processual, *fire&gas*, gestão de motores, entre outros, é essencial que não ocorram erros na migração por troca de registos ou pela ausência da configuração de alarmes, que poderá dar origem a incidentes processuais com alguma gravidade.

No diagrama da figura 5.15 é apresentado o diagrama sequencial dos passos necessários para efectuar a migração de um *PLC*.

Figura 5.15 - Diagrama de passos sequencial da passagem de *PLC* de *RS3* para *DeltaV*.

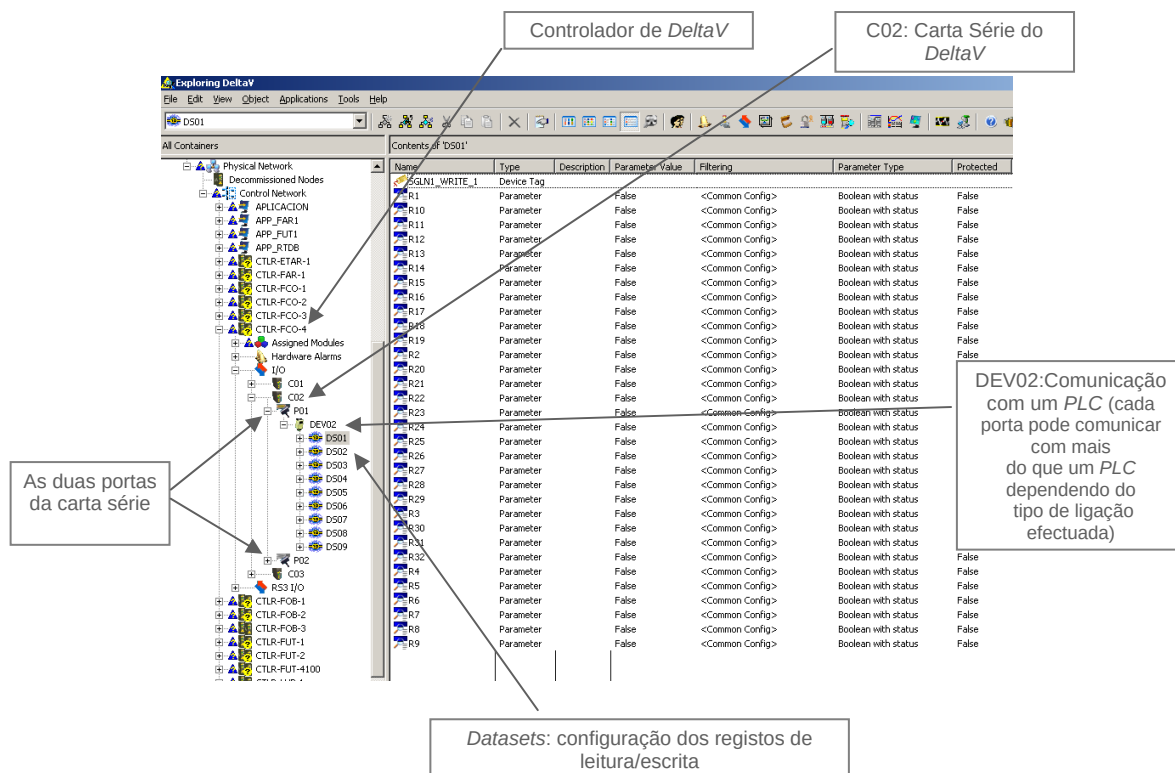


Figura 5.16 - Configuração dos registos agrupados em *datasets* no *DeltaV*.

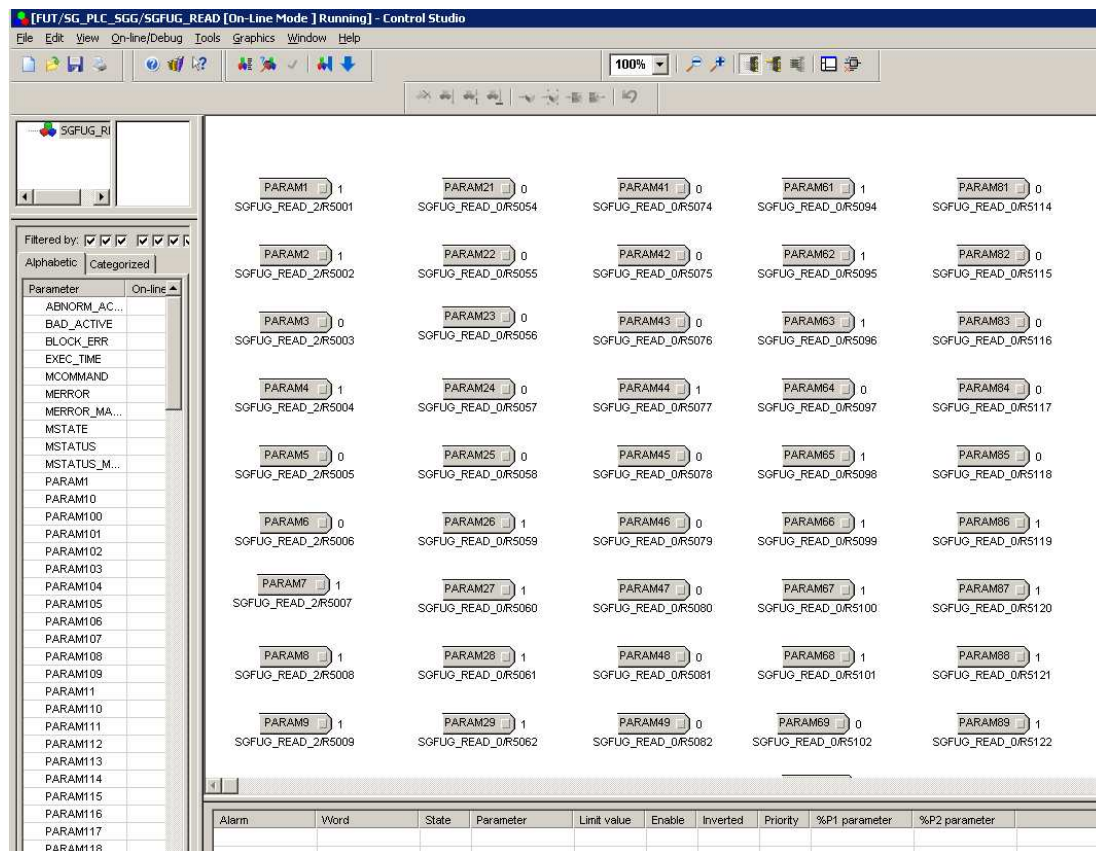


Figura 5.17 - Módulo base de leitura do *DeltaV* que lê parâmetros de um *dataset*.

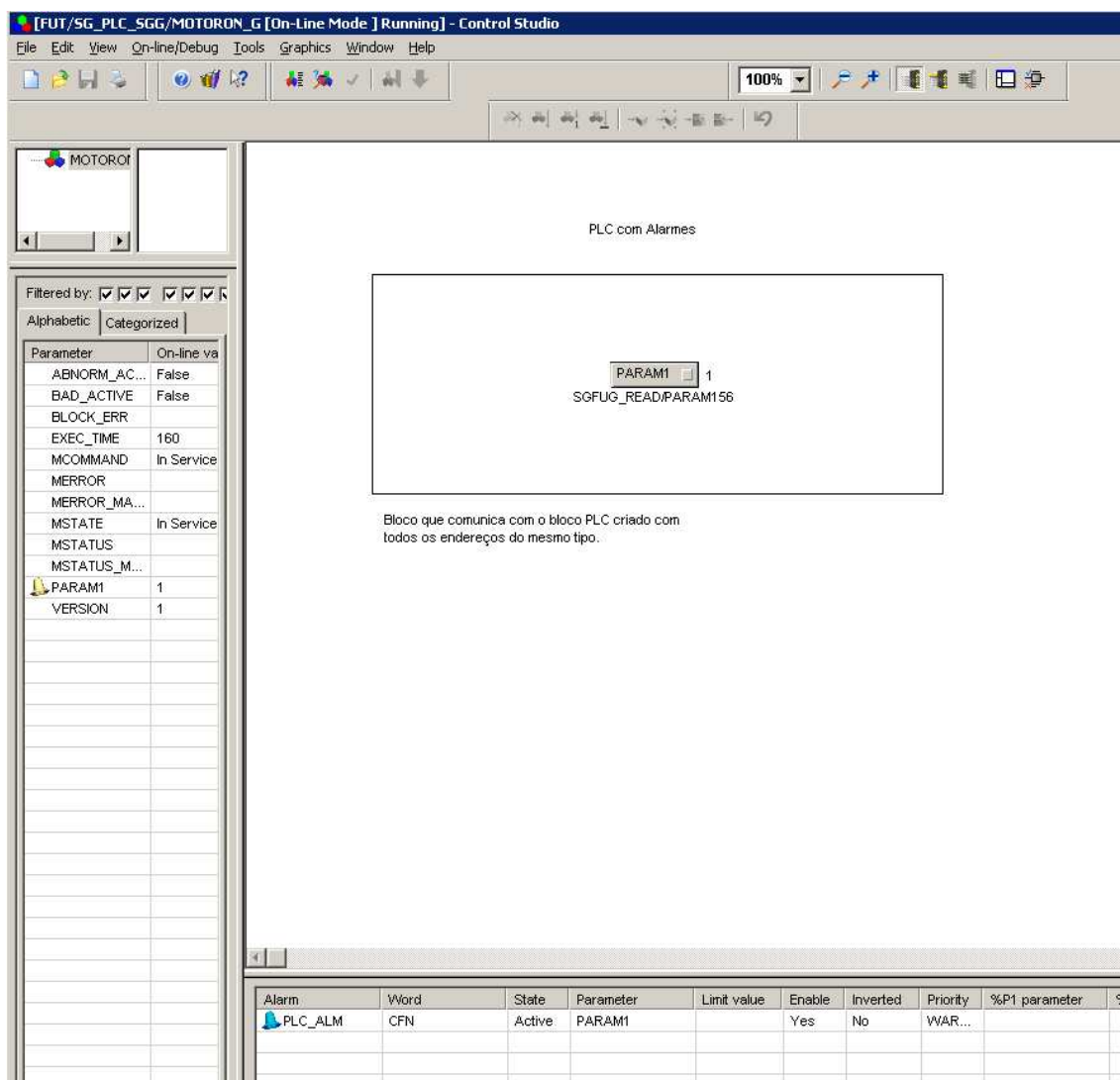


Figura 5.18 - Módulo individual (apenas um parâmetro) de *DeltaV* com alarmes para sinal digital.

Devido ao elevado número de passos e visto que o número de registos é habitualmente grande, na ordem das centenas, a migração é um processo lento e torna-se mais fácil a introdução de erros de configuração no *DeltaV*. A passagem de um *PLC* com, aproximadamente, 300 registos poderá demorar cinco dias úteis a configurar numa base de trabalho de 7 horas diárias. Esta duração não contempla a configuração da interface gráfica para visualização (sinópticos).

Este tipo de configuração manual é, como já foi referido, sujeito a um elevado número de erros. Na figura 5.19 mostra-se o método utilizado para os despistar e solucionar, sempre que possível. Após teste preliminares, os principais actores deste processo são os chefes de turno e operadores de painel experientes que detectam erros de animação da parte gráfica ou a ausência de alarmes em determinados *tags*.

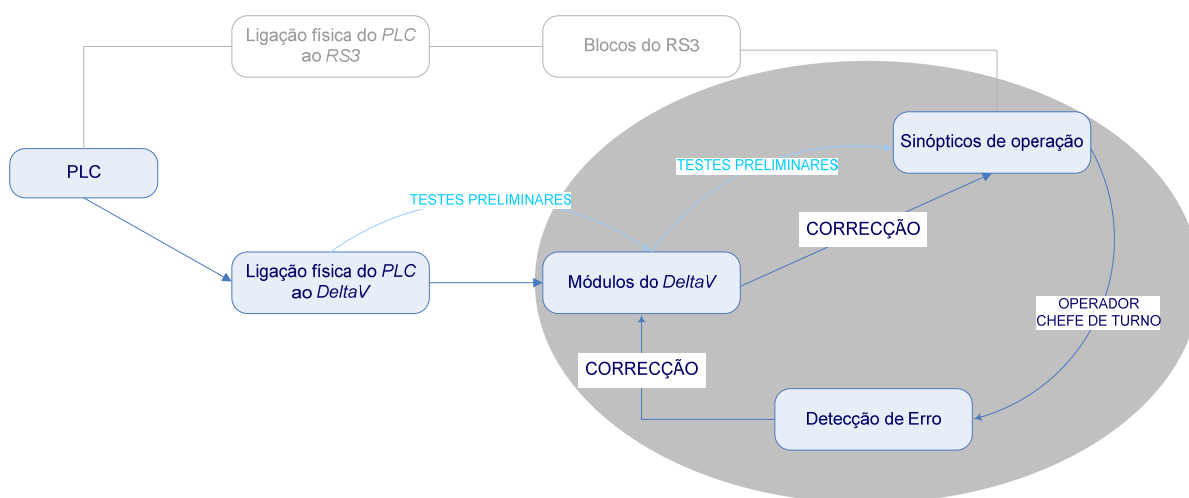


Figura 5.19 - Método para despistar e solucionar os erros na migração manual de PLC.

A ferramenta que se pretende conceber para efectuar a migração dos blocos *PLCB* desde o *RS3* para o *DeltaV*, deverá ser capaz de criar no controlador, carta e porta seleccionadas pelo utilizador o conjunto de *datasets* (figura 5.16) que abrangem o conjunto de blocos *PLCB* previamente seleccionados. Um *dataset* é o conjunto de registos *MODBUS* que se pretende configurar no sistema, encontrando-se agrupados em leituras e escritas. Após a criação dos *datasets* terão que ser criados todos os módulos do tipo base (figura 5.17) e os módulos individuais (figura 5.18) que poderão ou não ter alarmes associados. Esta análise cuidada do conjunto de módulos *PLCB* terá em linha de conta os seguintes parâmetros:

- *Block Type*,
- *Data Type*: discreto ou contínuo,
- Alarmes associados a cada *flag*,
- Descritivo associado a cada *flag*,
- *Read e*,
- *PLC Word Address*.

Os detalhes da especificação da solução proposta para a migração destes blocos são discutidos no capítulo seguinte.

5.3.2. Processo de Migração de Configuração de Válvulas e Bombas

Nas unidades processuais a migrar desde o *RS3*, existem inúmeros equipamentos, entre os quais diversos tipos de válvulas e bombas.

A engenharia de *software* destas unidades foi executada pela *Emerson Process Management* que utilizou configurações diferentes daquelas que foram adoptadas pela refinaria. É, assim, necessário efectuar um estudo dos blocos *DMC - Discrete Motor Controller* - (figura 5.20) do *RS3* de forma a ser possível implementar uma nova estratégia em *DeltaV*.

Esta nova estratégia deve contemplar as mesmas funcionalidades do bloco de *RS3* e, sempre que possível introduzir algumas melhorias, nomeadamente no que diz respeito à quantidade e qualidade da informação disponibilizada ao operador.

O bloco correspondente ao *DMC* do *RS3* em *DeltaV* é denominado por *DC - Device Control* (figura 5.22).

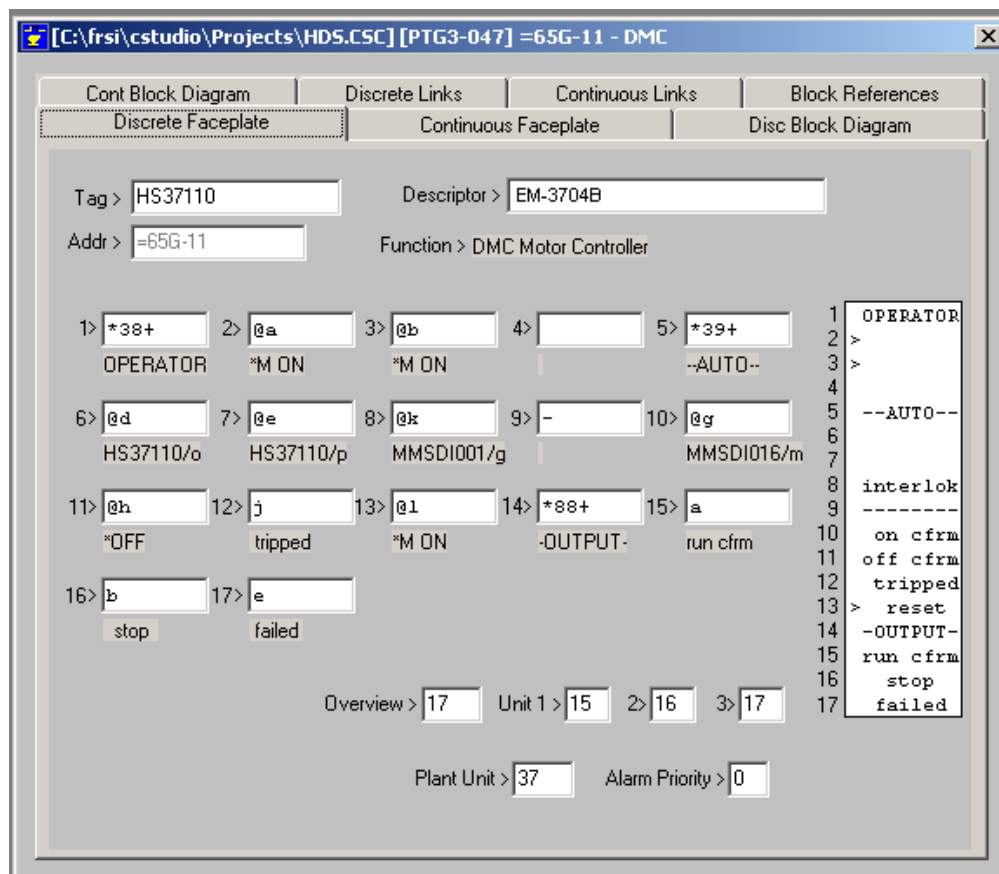


Figura 5.20 - Bloco *DMC* do *RS3* usado para válvulas e bombas.

Apesar do bloco *DMC* ser pré-configurado no *RS3*, existem vários *links* discretos (ecrã dos *Discrete Links*) que devem ser configurados para que ele fique perfeitamente operacional. Na tabela 5.1 estão todos estes *links* conjuntamente com um breve descritivo de cada um. Relativamente aos *links* contínuos este bloco contém cinco pré-configurados: *F*, *G*, *H*, *I* e *J*. Os restantes, de *A* a *E* e de *K* a *O* são opcionais para o configurador.

Tabela 5.1 - Links para a configuração de um bloco DMC [20].

Link	Aplicação
@a	Passo lógico a para arranque
@b	Passo lógico b para paragem (opcional)
@g	Sinal de confirmação de arranque
@h	Sinal de confirmação de paragem (opcional)
@d	Sinal para arranque automático (opcional)
@e	Sinal para paragem automática (opcional)

Após a configuração dos *links* anteriormente referidos, podem ser configuradas várias opções relativas à funcionalidade do bloco (figura 5.21), nomeadamente *interlocks*, a existência de parâmetros opcionais como o sinal de paragem, ou os sinais para arranque/paragem automáticos do equipamento.

No anexo 4 estão listadas todas as opções de configuração dos blocos DMC, bem como as funcionalidades de todos os links discretos e contínuos e passos lógicos, que também terão de ser contemplados no novo módulo de *DeltaV*.

Existem quatro modos possíveis para este bloco (*Block Mode*): local, manual, automático e *computer*. O modo local não permite que sejam efectuados comandos desde do DCS, ou seja, qualquer comando activado será ignorado e não será enviado para o equipamento; o modo manual permite a operação com ordens dadas pelo operador (arranque e paragem); o modo automático permite que o equipamento seja operado sob certas condições de funcionamento, por exemplo, o sinal de arranque será dado quando o nível de um tanque descer abaixo de uma determinada percentagem e o sinal de paragem accionado quando o referido nível estiver acima de um valor pré-definido; o modo *computer* permite a actuação deste equipamento por parte de uma outra interface ligada ao DCS.

CB CONTINUOUS FACEPLATE				27-May-92	09:52:59
Tag >	Desc >		Shed Mode>	NONE	>OUT>A>B>C>D
Addr =1A-01	Function >DMC	Motor Controller		SC>N	+-----+
INPUTS Value	Units	Tag	OUTPUT	A	.00
A >.00			>.00	B	.00
B >.00				C	.00
C >.00				D	.00
D >.00				100-	+-----+
				75-	+
INPUTS F	G			50-	+
Target >5.	>2.			25-	+
Value 0.	0.			0-	+-----+
Units secs	secs			OUT	.00
Func STRT TMR	STOP TMR				
Interlock >no		Retry >no			+-----+
Confirm Off >no		Security Lockup >no			
	Ignore Confirm On >no	Trip Delay>no	Block Mode>	LOCAL	
FULL				CONFIG 1	

Figura 5.21 - Bloco DMC com os parâmetros de configuração colocados por defeito.

Conforme referido anteriormente, o *DeltaV* possui um bloco de função com características semelhantes ao do *RS3*: o *DC* (figura 5.22). Os módulos que utilizam este bloco *DC* deverão fornecer, igualmente, informação ao operador do estado do equipamento, de eventuais *interlocks* activos e temporizações de operação.

Conjuntamente com a configuração do módulo em *DeltaV* será necessário fazê-lo acompanhar por um *faceplate* que possibilite a correcta operação do equipamento. O *faceplate* de *RS3* proveniente da configuração de *links* discretos (figura 4.8) será substituído por um novo *faceplate* que possibilite um maior grau de informação ao operador, de forma a proporcionar maior autonomia na condução das unidades processuais.

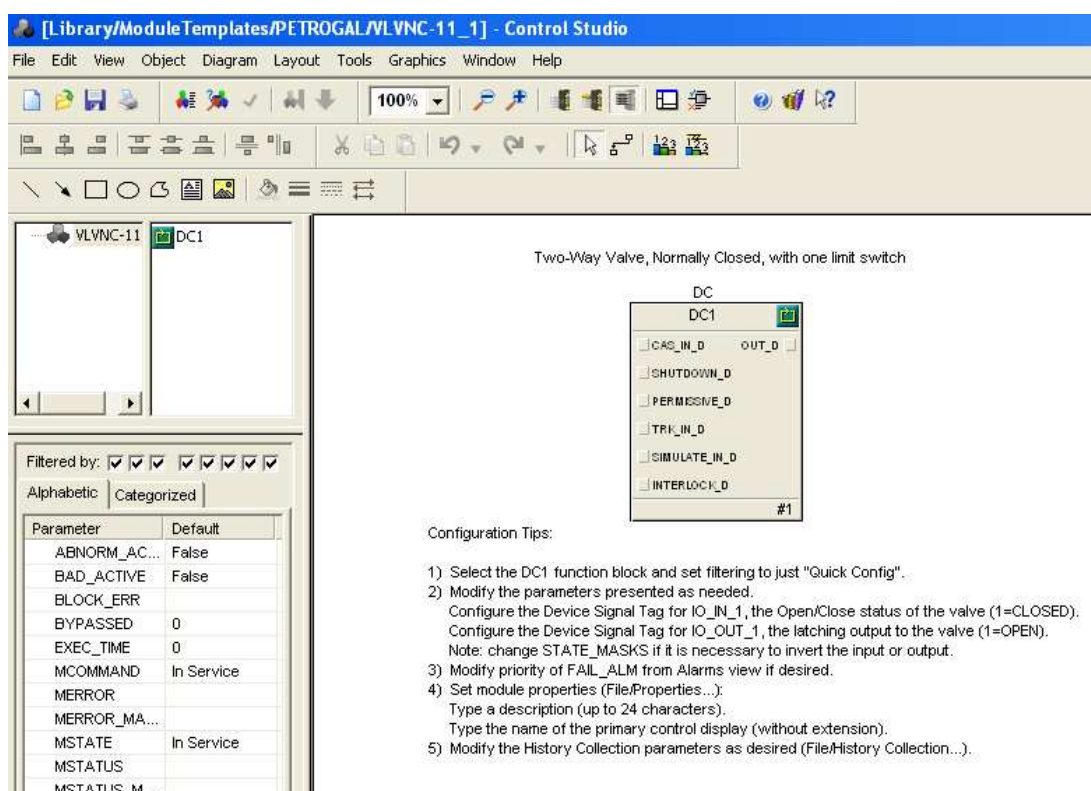


Figura 5.22 - Módulo com o bloco de função *DC* do *DeltaV* usado para válvulas e bombas.

Na criação do *faceplate* de operação e do *faceplate* de detalhe recorre-se à aplicação *DeltaV Operate*, sendo essencialmente constituída por uma barra de ferramentas, na parte superior da aplicação, uma área de sinópticos e a área inferior dedicada à visualização e reconhecimento de alarmes (figura 5.23).

A barra de ferramentas permite aceder aos diversos sinópticos de operação disponíveis no PC, aceder a tendências, a listagens de alarmes, a *faceplates* e *faceplates* de detalhe para um dado *tag* introduzido pelo operador, obter ajuda referente a diversos tópicos pela disponibilização de *Books Online* do *DeltaV* e do *RS3*, alterar o acesso através da alteração do utilizador e respectiva *password*, entre outras funcionalidades. Esta barra pode ser configurada para ir ao encontro das especificações do utilizador.

A barra de alarmes apresenta os detalhes do último alarme mais prioritário, classificado como alarme crítico, nomeadamente a data e hora de ocorrência, o nome do *tag* e parâmetro em alarme (um *tag* pode ter diversos alarmes) e respectivo descritivo.

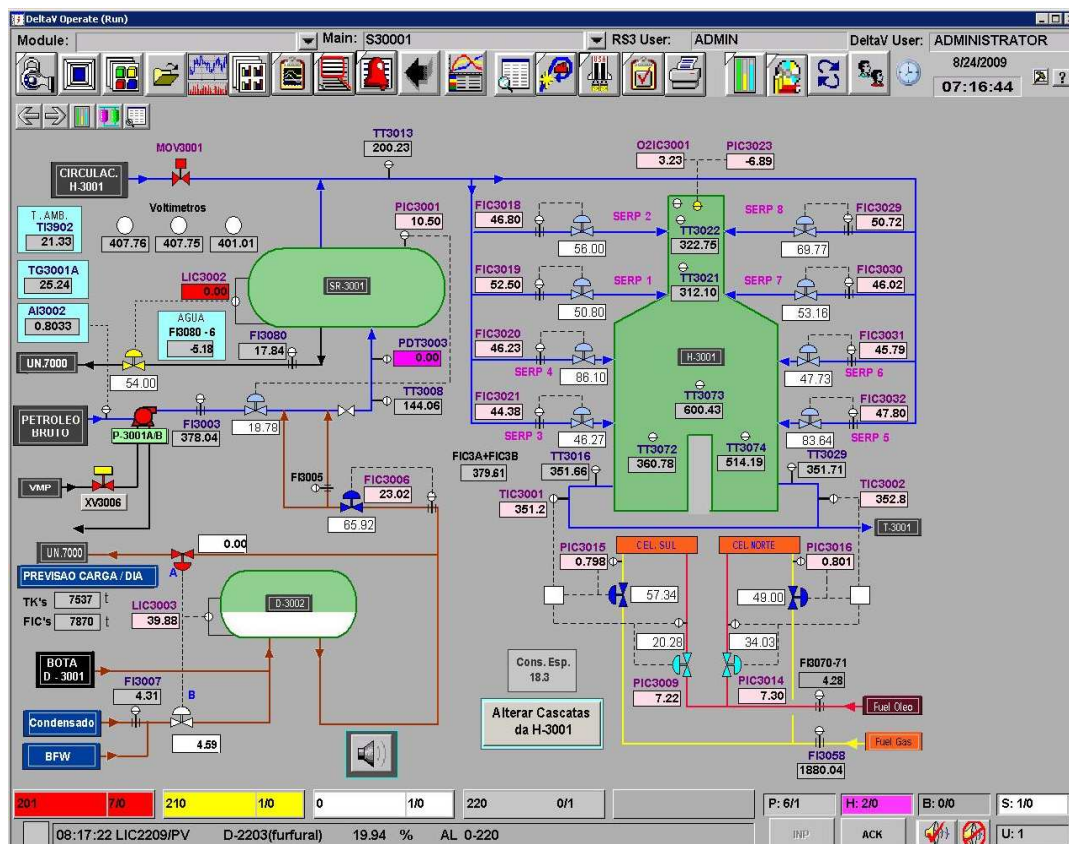


Figura 5.23 - Sinóptico de operação usando a aplicação *DeltaV Operate*.

O *DeltaV Operate* permite, em modo de configuração, o desenvolvimento de toda a interface gráfica de operação (sinópticos) que acedem, conforme a taxa de refrescamento definida, à base de dados do *RS3* e do *DeltaV* dependendo de onde se encontra configurado o *tag*. A operação processual é igualmente efectuada recorrendo à aplicação referida mas agora no modo denominado por *run*. É ainda possível desenvolver e aceder a todos os tipos de *faceplates* e caixas de diálogo que são apresentadas aos operadores nas situações pretendidas. A figura 5.24 apresenta os *faceplates* de operação para válvulas e motores *standard* do *DeltaV*.

Da mesma forma que na criação do novo módulo de gestão destes equipamentos se irá utilizar como base um bloco pré-configurado, também no desenvolvimento de *faceplates* serão usados os da figura 5.24 como base de desenvolvimento.



Figura 5.24 - *Faceplate* para o bloco DC e respectivo *faceplate* de detalhe.

Capítulo 6

Especificação da Solução Proposta

Este capítulo apresenta a análise de requisitos e a concepção estrutural da arquitectura de funcionamento do programa informático de conversão dos blocos *PLCB* para *DeltaV*.

Os blocos de comando/monitorização de válvulas e bombas serão também analisados de forma profunda para especificar uma solução adequada à sua utilização no *DeltaV*. A operação destes equipamentos usando *faceplates* e *faceplates* de detalhe serão igualmente especificados e desenvolvidos.

6.1. Especificação

A especificação de uma aplicação é de vital importância para o seu pleno funcionamento. Na figura 6.1, em que se mostram as etapas do processo de desenvolvimento de uma aplicação, é possível identificar a primeira etapa deste processo: a definição de requisitos. É a partir desta etapa que as subseqüentes poderão ser correctamente desenvolvidas, nomeadamente o desenho da aplicação e o desenvolvimento e integração de sub-aplicações sempre que necessárias. Após a instalação e testes de validação poderá ser necessário proceder à sua desinstalação quando esta deixa de ser válida para o sistema.

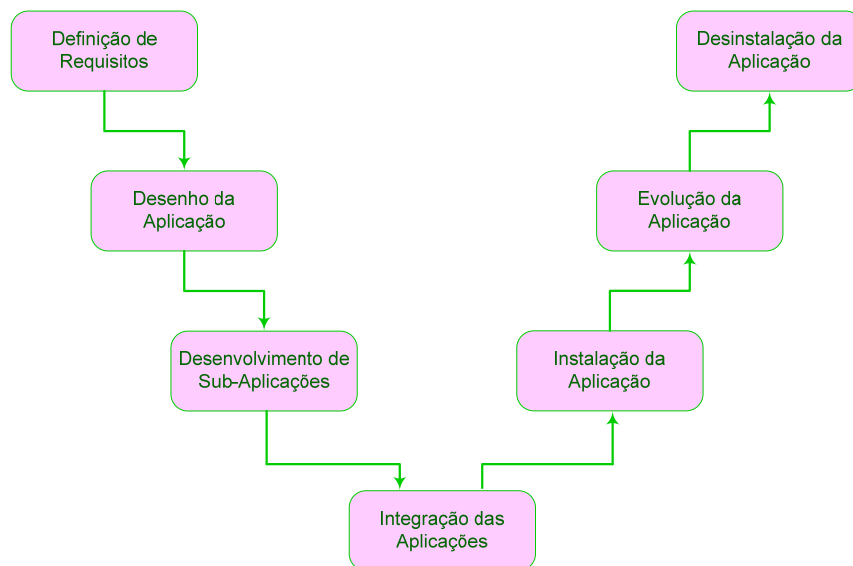


Figura 6.1 - Engenharia de processo de uma aplicação [21].

6.1.1. Requisitos

Como anteriormente referido, o ponto de partida para o desenvolvimento de uma aplicação é o processo de identificação dos requisitos para ser possível iniciar o seu desenho. O estudo da necessidade de desenvolvimento de sub-aplicações é também vital pois estas terão que ser testadas individualmente mas também são necessários testes de integração das mesmas em busca de incompatibilidades e eventuais incongruências.

O processo seguido para o desenvolvimento da aplicação é mostrado na figura 6.2. Várias versões foram construídas até à versão final da aplicação visto ter existido um “*wicked problem*”: problema complexo onde existem diversas variáveis relacionadas e para o qual não existe uma especificação definitiva e final da solução.

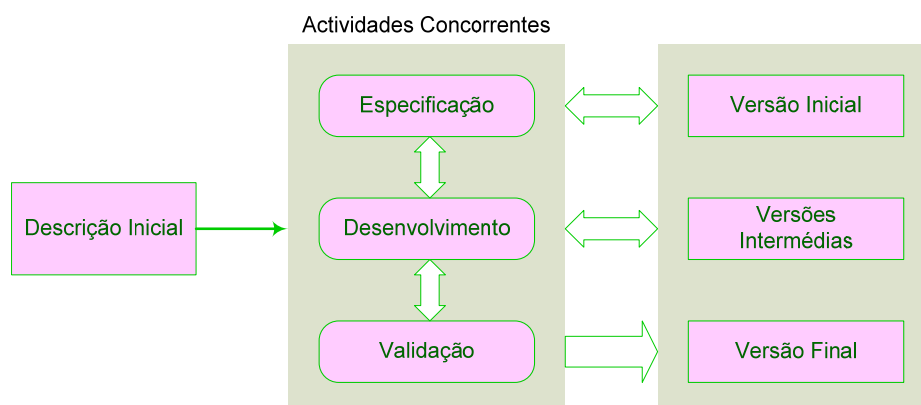


Figura 6.2 - Desenvolvimento evolutivo de uma aplicação [21].

Para correcção de erros foi seguido o método da figura 6.3. Após a conclusão da aplicação foram identificados os erros e foram sendo corrigidos até à eliminação dos mesmos, dando origem a uma nova versão da aplicação.

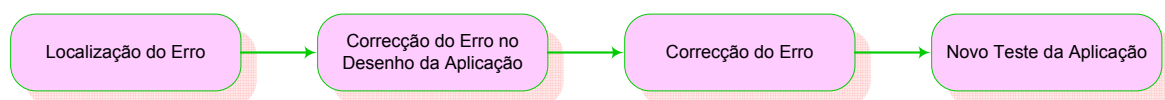


Figura 6.3 - Processo de correcção de erros [21].

A análise de requisitos para a construção da aplicação que permita efectuar a migração dos blocos *PLCB* do *RS3* para o *DeltaV* é apresentada na secção seguinte.

Relativamente ao estudo da forma mais adequada para a migração das lógicas de controlo de válvulas e motores, a análise de requisitos é efectuada na secção 6.1.1.2. Neste caso não foi desenvolvida uma aplicação de *software* mas sim módulos base para efectuar a migração de acordo com a nova filosofia de *software* do *DeltaV*.

6.1.1.1. Análise de Requisitos - Migração de Configurações de *PLC*

A aplicação tem por objectivo a criação de *datasets* de leitura e/ou escrita conforme os blocos *PLCB* do *CS for RS3* e a criação de módulos para ler ou escrever nesses mesmos *datasets*. Conforme referido no capítulo anterior, um *dataset* é um conjunto de registos *MODBUS* agrupados em leitura e escrita sendo que, cada *dataset*, poderá ler/escrever um máximo de 100 registos.

O ficheiro gerado na criação de *datasets* é do tipo *.TXT* e, portanto, é necessário utilizar um ficheiro de especificação de formato (*.FMT*). Assim, terá que ser criado um ficheiro de especificação de formato no *DeltaV* que permita a importação usando a opção *User Defined Format* (ver figura 5.6). Este formato contém os dados essenciais para a criação de *datasets*. O nome do ficheiro de especificação de formato é: *Serial_Dataset.fmt* (Anexo 5). A não utilização de um ficheiro de formato de importação directa no *DeltaV* prende-se com o facto de serem necessários dados adicionais não disponíveis no *CS for RS3* e visto serem configurações muito simples de acrescentar e alterar manualmente. Assim, foi seleccionada a opção de utilização de um novo formato criado especificamente para este fim, embora esteja sempre dependente de configurações manuais prévias à importação do ficheiro.

Nas figuras 6.4 a 6.7 são apresentados os quatro sub-tipos de blocos *PLCB*: dois tipos básicos de blocos *PLCB* relativos à direcção da informação: leitura (*Read = normal*) e escrita (*Read = No*); cada um deles é ainda subdividido em outros dois (*Data Type*): *flags* (para valores discretos) ou *U Int* (para valores analógicos). A figura 6.8 permite visualizar esta divisão de uma forma mais simplificada.

Tag/Address | Block References

I/O BLOCK CONFIGURATION

Address > =33D096

Block Tag > SGD-SG16

Auto Lock > Yes

Source >

Addr >

Block Type > PLCB

Data Type > Flags

Sample Time > 1 S

Sim Tag >

Addr >

Port > B PLC number > 176

Read > normal

PLC Word Address > 5098

Flags	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p
a:	CFRB-D				e:	SPCP-D			i:	RSRA-D			m:	S2-D		
b:	COP-D				f:	ST40201H-D			j:	LAH40311-D			n:	FCV40201-D		
c:	SPAC-D				g:	ST40201HH-D			k:	LAL40314-D			o:	FIM_DE_PURGA-D		
d:	SPC-D				h:	RSND-D			l:	LOP-D			p:	L4-C_D		

Priority > 0 Plant Unit > 120

Figura 6.4 - Parâmetros necessários dos blocos *PLCB* para a fase I - Leitura tipo *Flags*.

Tag/Address | Block References

I/O BLOCK CONFIGURATION

Address > =105G001

Block Tag > LIT25518A

Auto Lock > No

Source >

Addr >

Block Type > PLCB

Data Type > U Int

Sample Time > 1 S

Sim Tag >

Addr >

Port > A PLC number > 1

Read > normal

PLC Word Address > 40001

Priority > 0 Plant Unit > 250

Figura 6.5 - Parâmetros necessários dos blocos *PLCB* para a fase I - Leitura tipo *U Int*.

Tag/Address | Block References

I/O BLOCK CONFIGURATION

Address > =105G084

Block Tag > SCHNEIDER4

Source > DOSCHNEIDER4

Addr > =105G-44

Write Filter > normal

Block Type > PLCB

Data Type > Flags

Sample Time > 1 S

Sim Tag >

Addr >

Read > no

Auto Lock > No

PLC Write Mask > FFFFx

Port > B PLC number > 1

PLC Word Address > 40005

Flags	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p
a:					e:				i:	UXP2559A_stop			m:	UXP2555B_stop		
b:					f:				i:	UXP2558_stop			n:	UXP2555B_start		
c:					g:				k:	UXP2557_stop			o:	UXP2555A_stop		
d:					h:	UXP2559B_stop			l:	UXP2556_stop			p:	UXP2555A_start		

Priority > 0 Plant Unit > 250

Figura 6.6 - Parâmetros necessários dos blocos *PLCB* para a fase I - Escrita tipo *Flags*.

Tag/Address | Block References

I/O BLOCK CONFIGURATION

Address > =33C036

Block Tag > FIQ4401-SP

Source > FIQ4401/G

Addr > =33C-18

Write Filter > normal

Block Type > PLCB

Data Type > U Int

Sample Time > 1 S

Sim Tag >

Addr >

Read > no

Auto Lock > No

PLC Write Mask > FFFFx

Port > B PLC number > 1

PLC Word Address > 42751

Priority > 0 Plant Unit > 44

Figura 6.7 - Parâmetros necessários dos blocos *PLCB* para a fase I - Escrita tipo *U Int*.

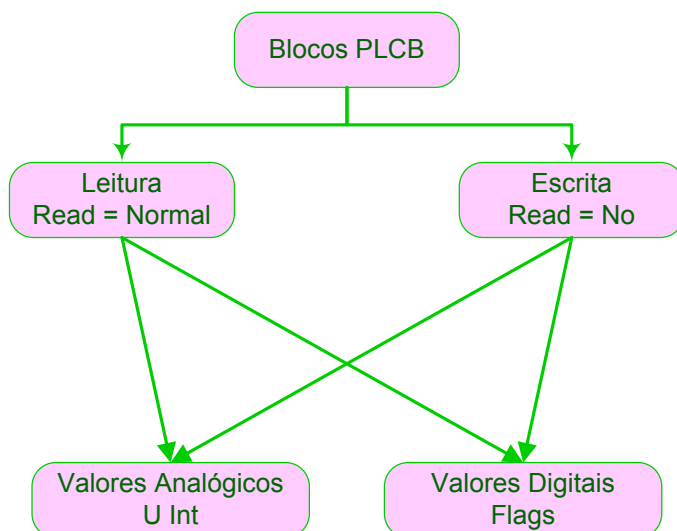


Figura 6.8 - Divisão dos blocos *PLCB* para a fase I.

Para criar o ficheiro texto é igualmente necessário conhecer o tipo de registos *MODBUS* em questão. Pela análise do parâmetro *PLC Word Address* dos blocos *PLCB* e a sua transposição para o *DeltaV* temos os seguintes tipo:

- *Coils*,
- *Holding Register*,
- *Input Status*,
- *Input Register*.

A tabela 6.1 resume a gama de endereços utilizados no *DeltaV* e a sua denominação, sendo esta essencial para a produção do ficheiro *.TXT*.

Na criação de *datasets*, o *DeltaV* não trabalha com valores absolutos de registos mas sim valores relativos aos apresentados na tabela 6.1, ao qual denominam *offset*. Por exemplo, será migrado o conjunto de registos 40101 ao 40150. Sendo este conjunto de registos superior a 40001, é classificado como *Holding Registers*. Logo, são necessários ler/escrever 50 registos e o *offset* será 100 visto que a diferença para o registo inicial desta gama é exactamente 100 (40101-40001=100).

No Anexo 6 são apresentados todos os parâmetros do ficheiro de texto a ser gerado.

Tabela 6.1 - Endereços *MODBUS* e respectiva descrição no *DeltaV*.

Endereços <i>MODBUS</i>	Descrição
1 ... 10000	<i>Coils</i>
10001 ... 30000	<i>Input Status</i>
30001 ... 40000	<i>Input Register</i>
40001 ...	<i>Holding Register</i>

Criados os *datasets*, são necessários criar os módulos de *DeltaV* a partir de três fontes de informação: ficheiro *.TXT*, de dados provenientes do *CS for RS3* e de dados introduzidos pelo utilizador. A necessidade de criar módulos prende-se com o facto dos registos estarem apenas disponíveis ao configurador, visto estarem apenas configurados na carta/porta, ou seja, estão disponíveis em *datasets*. Para que o valor do registo fique disponível ao operador é necessário criar um módulo para cada um dos registos, com um *tag* único e que possa ser referenciado de forma inequívoca.

Foram, assim, criados quatro tipos básicos de módulos denominados por *landing modules*:

- *Landing module* de leitura para registos consecutivos do tipo *Flags - READ_1* com um máximo de 10 parâmetros que estarão ligados a 10 registos provenientes dos *datasets* criados (figura 6.9);
- *Landing module* de leitura para registos não consecutivos do tipo *Flags* e consecutivos e não consecutivos do tipo *U Int - READ* com um máximo de 50 parâmetros que estarão ligados a 50 registos provenientes dos *datasets* criados (figura 6.10);
- *Landing module* de escrita para registos consecutivos do tipo *Flags - WRITE_1* com um máximo de 10 parâmetros que estarão ligados a 10 registos que estarão ligados aos *datasets* criados (figura 6.11);
- *Landing module* de escrita para registos não consecutivos do tipo *Flags* e consecutivos e não consecutivos do tipo *U Int - WRITE* com um máximo de 50 parâmetros que estarão ligados a 50 registos que estarão ligados aos *datasets* criados (figura 6.12);

A denominação *landing modules* deve-se ao facto deste módulos compreenderem um conjunto de registos que poderão ou não ser utilizados individualmente em módulos a jusante. A criação destes módulos permite gastar um número mais reduzido de licenças de *software* do que a colocação directa de leitura/escrita de registos a partir do *dataset*, ou seja, por cada módulo individual que lê ou escreve num registo de um dado *dataset* é gasta uma licença. Se se utilizar um *landing module*, independentemente do número de registos que estejam a ser lidos ou escritos dentro desse módulo solicitará uma única licença. Todos os módulos a jusante deste não gastarão qualquer licença, visto não estarem ligados directamente a um *dataset* mas sim a um *landing module*.

A denominação de registos consecutivos e não consecutivos está relacionada com o parâmetro *PLC Word Address* dos blocos *PLCB*. No caso do primeiro endereço ser 10050 e no endereço do bloco seguinte ser 10051 o endereçamento será consecutivo, caso seja 10067 será classificado como não consecutivo.

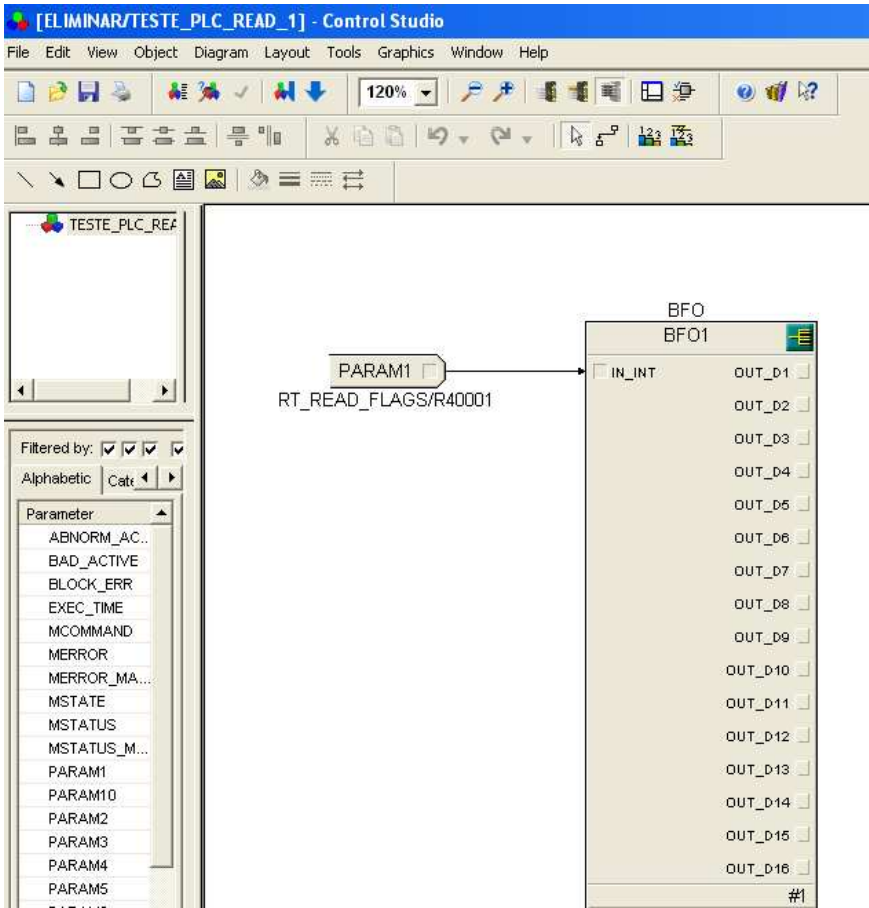


Figura 6.9 - Landing Module tipo READ_1 - número máximo de parâmetros é 10.

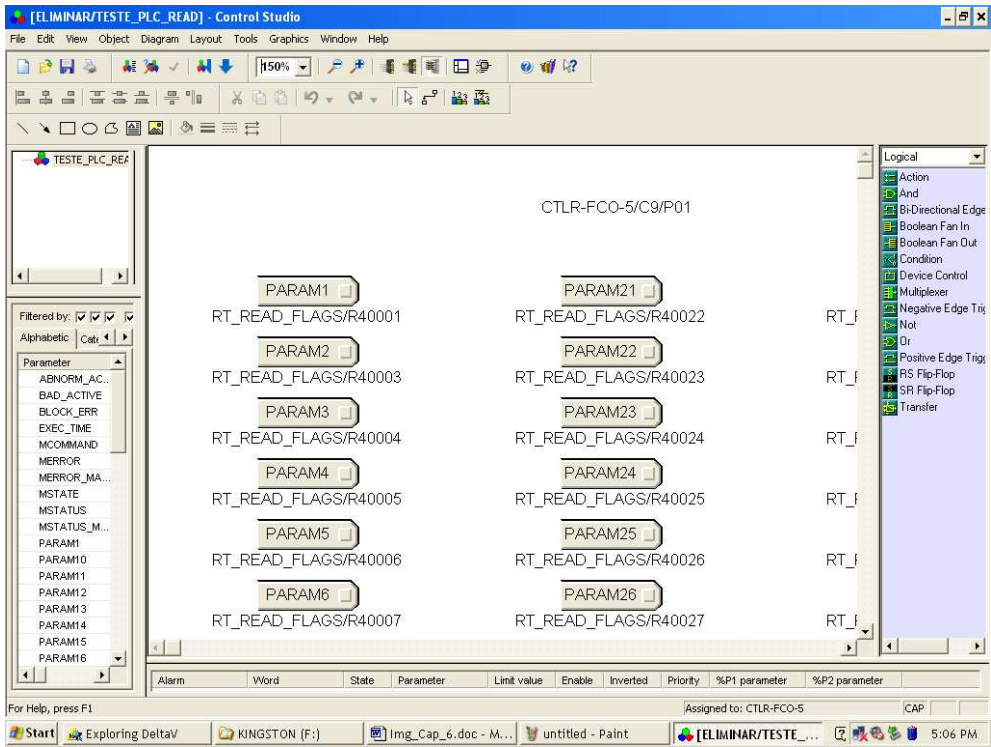


Figura 6.10 - Landing Module tipo READ - número máximo de parâmetros é 50.

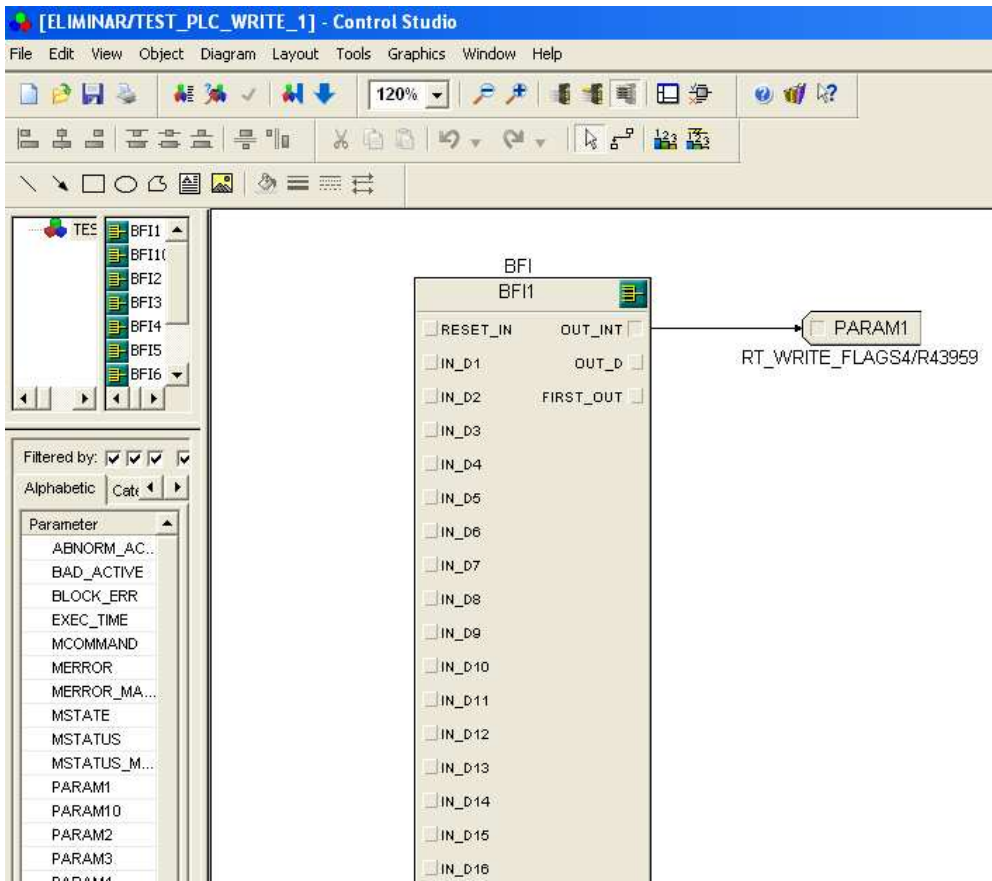


Figura 6.11 - Landing Module tipo *WRITE_1* - número máximo de parâmetros é 10.

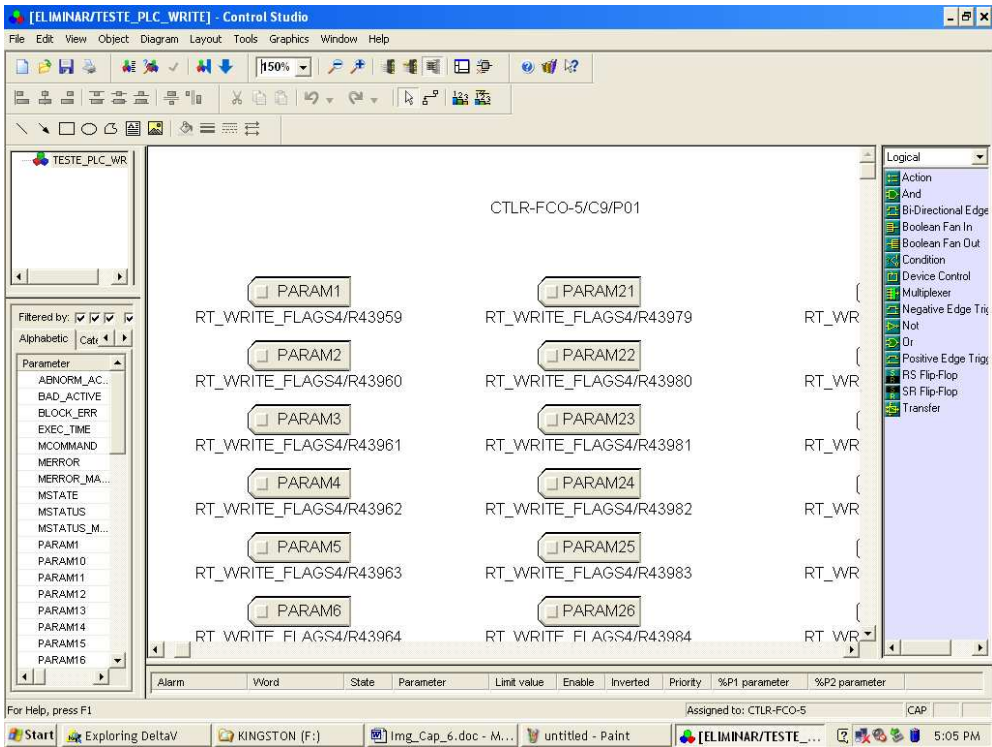


Figura 6.12 - Landing Module tipo *WRITE* - número máximo de parâmetros é 50.

A existência de quatro tipos de *landing modules* está relacionada com o facto dos registos serem ou não consecutivos e do *Data Type* envolvido. Quando temos registos consecutivos que são do tipo *flags* significa que no mesmo registo está a receber ou a enviar um conjunto de 16 *bits* e, portanto, este valor terá que ser “descodificado”. No *DeltaV* existem blocos de função já preparados para efectuar este processo: *Boolean Fan In (BFI* - para leituras) e *Boolean Fan Out (BFO* - para escritas). Nas figuras 6.9 e 6.11 é visível a utilização destes dois tipos de blocos de função tanto para escritas como para leituras.

Após este primeiro processo são necessários criar os módulos individuais que vão beber a sua referência no respectivo *landing module*, verificar que dados adicionais são necessários recolher do *CS for RS3* tais como *tag*, tipo de alarme, quando deverá alarmar e descritivo de alarme e que dados são necessários do utilizador. Estes módulos serão utilizados nas animações das interfaces gráficas em detrimento dos *landing modules*. Foram desenvolvidos dois tipos de módulos base para leituras:

- *DI* para leitura de valores discretos: 0 ou 1/*On* ou *Off* (figura 6.13) e,
- *AI* para leitura de valores analógicos (figura 6.14).

No anexo 7 são descritos pormenorizadamente todos os parâmetros envolvidos na aplicação desta fase de criação de módulos.

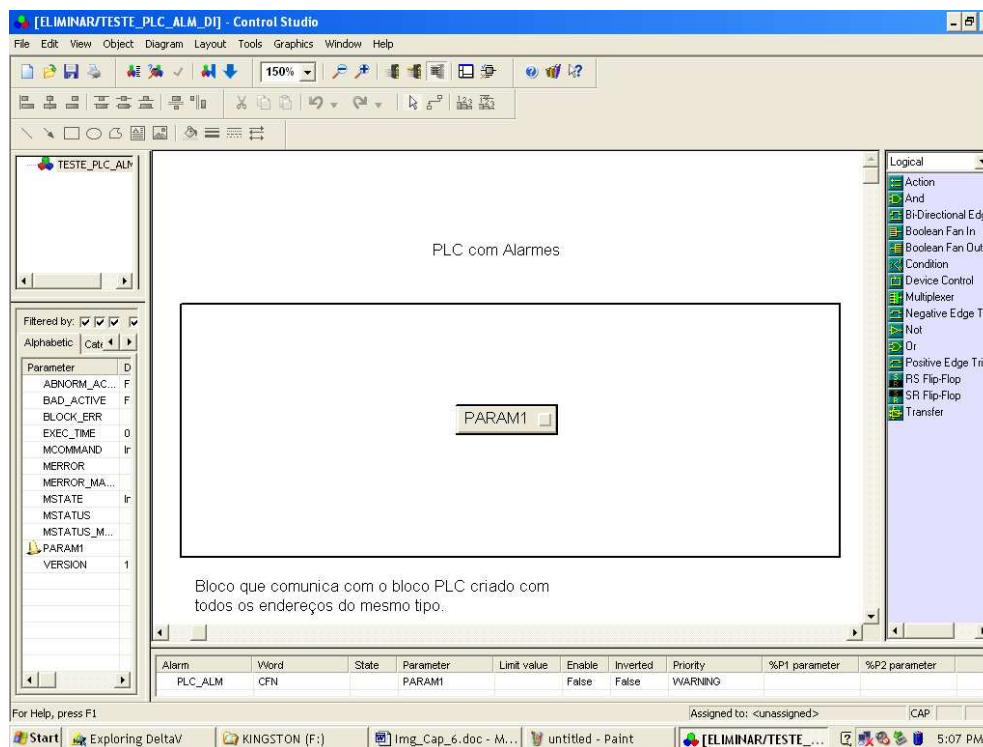


Figura 6.13 - Módulo individual tipo *DI* para leituras discretas.

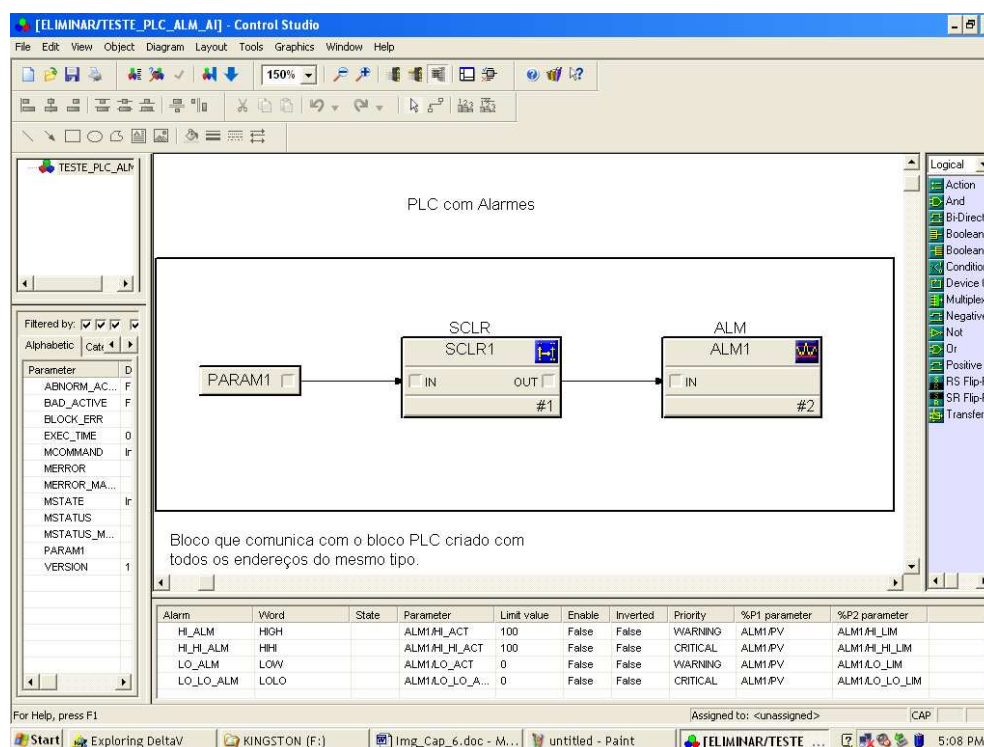


Figura 6.14 - Módulo individual tipo AI para leituras analógicas.

Para as leituras analógicas é utilizado um bloco de função adicional o *Scaler* (SCLR). Este bloco permite efectuar mudanças de escala: de uma escala original para o valor de entrada para outra escala de saída. É frequente os configuradores dos registos num PLC enviarem para o DCS valores não escalonados originando valores não apropriados processualmente visto terem que ser convertidos para outra escala. Caso este bloco não seja necessário é apenas necessário configurar a escala do valor de entrada igual à escala do valor de saída.

No caso dos registos de escrita apenas serão criados os *landing modules* visto que a forma de escrita nestes registos depende muitas vezes de um conjunto de acções de diversos blocos do RS3 ou depende de um comando do operador. Como habitualmente não são muitos os blocos e é usual colocar o comando desde a interface gráfica directamente no *landing module* não será necessário analisar qualquer bloco adicional para este caso.

6.1.1.2. Análise de Requisitos - Configuração de Válvulas e Bombas

A análise cuidada do bloco DMC do RS3 efectuada no capítulo 5 permite, na análise de requisitos, recolher um conjunto de funcionalidades que terão que ser respeitados na migração destes blocos para o DeltaV, nomeadamente:

- Sinal de confirmação de abertura/arranque,
- Sinal de confirmação de fecho/paragem,
- Comando de abertura/arranque,

- Comando de fecho/paragem,
- Sinal para abertura/arranque automática(o) (caso exista),
- Sinal para fecho/paragem automático(a) (caso exista),
- Sinal de *interlock* do equipamento (caso exista) e,
- Sinal de *shutdown* do equipamento (caso exista).

Foi equacionado o desenvolvimento de uma aplicação que permitisse efectuar a migração de forma automática, mas diversos factores conduziram num sentido diferente:

- Número reduzido de blocos a migrar comparativamente aos utilizados no controlo analógico,
- Grande variabilidade da lógica para arranque e paragem automáticos e,
- Diferenças inerentes aos blocos de função do *RS3* e do *DeltaV*.

Estes factores, essencialmente o primeiro e o último, permitiram orientar este processo no sentido da criação de um módulo *standard* em *DeltaV* perfeitamente adequado às necessidades de todos os blocos *DMC* do *RS3*, e criar um *faceplate* para operação e um *faceplate* de detalhe com parâmetros adicionais para que o operador possa obter o máximo de informação do estado do equipamento e restrições ao seu funcionamento.

6.1.2. Solução Proposta

Após a análise de requisitos para o desenvolvimento da aplicação e para a criação de novos módulos para válvulas e bombas no *DeltaV*, é apresentada, nas secções seguintes, a solução proposta para cada um dos problemas.

6.1.2.1. Migração da Configuração de PLC

A aplicação desenvolvida está, conforme referido anteriormente, dependente de duas fontes de dados para gerar todos os ficheiros necessários: do *CS for RS3* e do utilizador.

A recolha dos dados pretendidos dos blocos *PLCB* desde o *CS for RS3* é efectuada recorrendo à biblioteca de objectos *COM* da ferramenta *Control Studio*, que permite seleccionar blocos e identificar os parâmetros pretendidos nesses mesmos blocos. O papel desta biblioteca é fornecer um conjunto de objectos dinâmicos que podem ser utilizados em ambientes de desenvolvimento diferentes do original, sem haver necessidade de conhecer a forma como estão implementados no sistema de base. O formato da base de dados do *CS for RS3* é *.CSC*.

Todos os módulos *DeltaV* necessários foram criados previamente ao desenvolvimento da aplicação: o ficheiro de especificação de formato, os quatro tipos *landing modules* e os módulos *DI* e *AI*. A ferramenta utilizada foi o *Control Studio* do *DeltaV* e a sua biblioteca de blocos de funções *standard* (ver anexo 2).

Inicialmente é gerado um ficheiro de texto e, posteriormente, são gerados ficheiros de configuração de *DeltaV*.

O princípio de funcionamento da aplicação é ilustrado na figura 6.15.

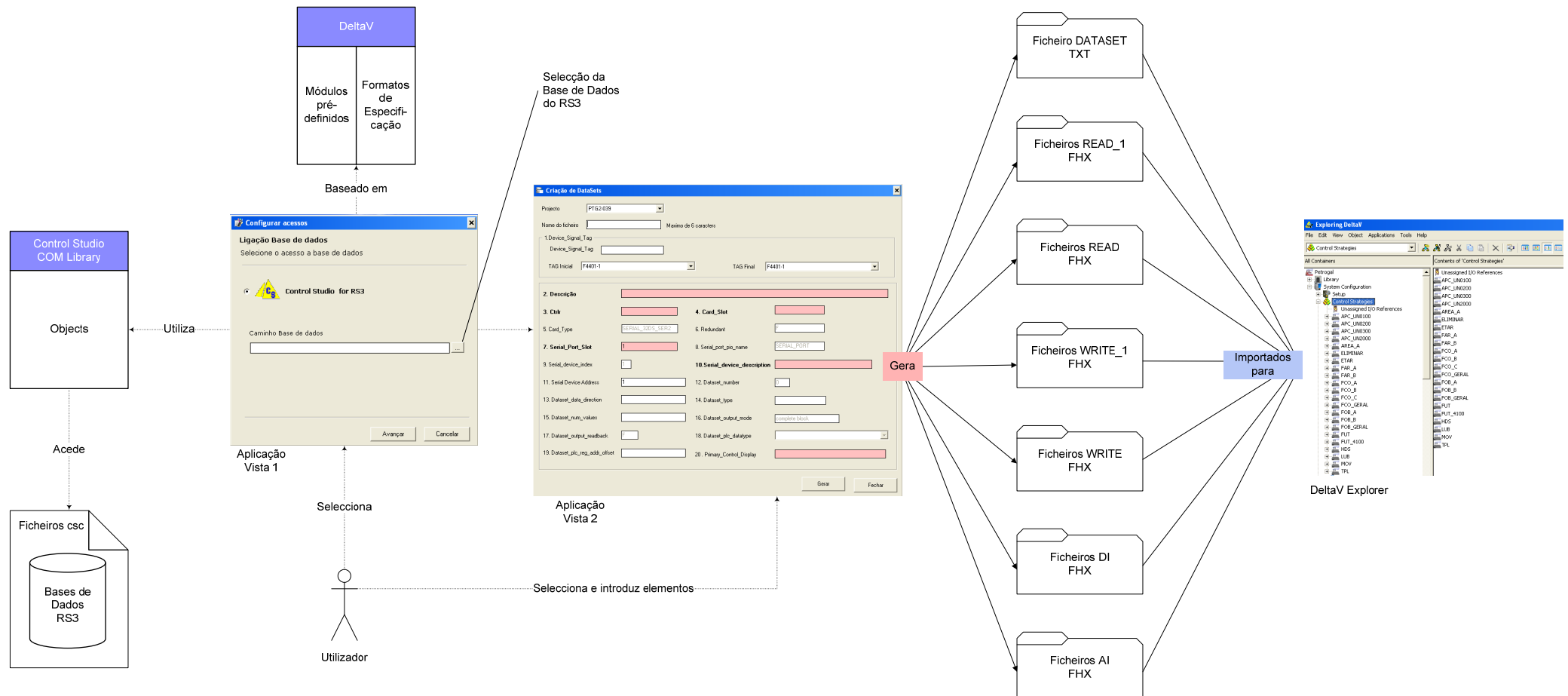


Figura 6.15 - Princípio de funcionamento da aplicação de migração de blocos *PLCB*.

A aplicação denominada *MG_PLC* (migração de blocos *PLC*) para efectuar a migração de configurações de blocos *PLCB* do *RS3* para o *DeltaV* foi dividida em duas fases:

- **Fase I:** Criação de *Datasets* e,
- **Fase II:** Criação de módulos no *DeltaV*.

A divisão da aplicação em duas fases distintas, mas absolutamente complementares, teve como base três aspectos fundamentais:

- Grau de complexidade: a aplicação como um todo tornou-se demasiado complexa ao nível da programação,
- Definição de metas a cumprir e,
- Estabelecimento de um prazo para a versão final.

Da figura 5.15, do capítulo 5, foi apresentado o diagrama sequencial dos passos necessários para efectuar a migração dos blocos desde o *RS3* para o *DeltaV*. Este diagrama é apresentado na figura 6.16 reestruturado e dividido nas duas fases complementares.

A configuração da carta e da porta no *DeltaV* exigem dados não disponíveis no *CS for RS3*, estando apenas disponíveis no sistema *online* ou por inspecção directa no *hardware* do *RS3* (verificação do tipo de carta utilizada para comunicação série). No entanto, é muito fácil e rápida a sua configuração no *DeltaV*, não sendo necessário empregar esforço adicional para o efectuar automaticamente, ou seja, o ganho em fazê-lo seria praticamente nulo.

A fase I contempla a área referente à configuração de registos de *PLC* na carta e porta pré-definidos e a fase II é referente à configuração de módulos de *DeltaV*, tanto para registos de leitura como para registos de escrita do/no *PLC*. Só após conclusão da fase I é que foi possível iniciar a fase II, visto que esta última é dependente da anterior.

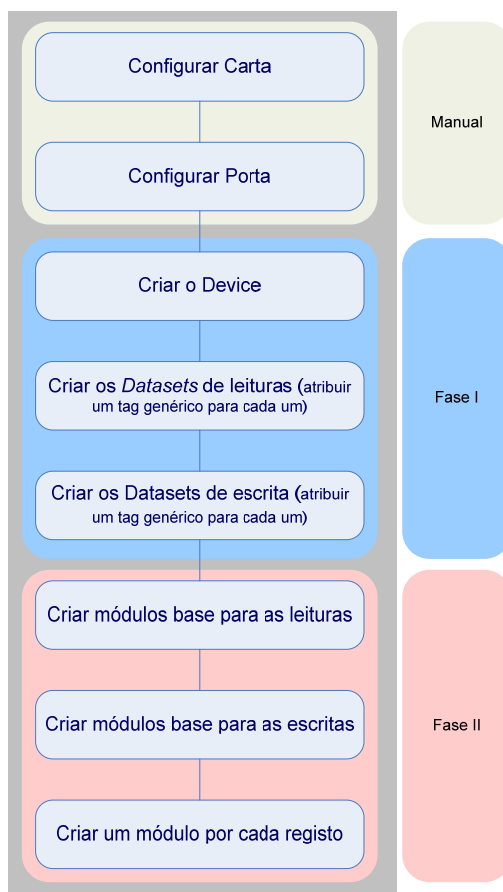


Figura 6.16 - Divisão da aplicação em duas fases complementares.

6.1.2.2. Modelação da Aplicação em *UML*

Para especificar a solução proposta para a aplicação utilizou-se a linguagem *UML - Unified Modelling Language*. Esta linguagem utiliza uma notação padrão para especificar, construir, visualizar e documentar sistemas de informação orientados por objectos [23].

A utilização da *UML* na especificação da aplicação prende-se com a abrangência e simplicidade dos conceitos que esta linguagem utiliza, permitindo integrar aspectos de natureza organizacional e aspectos de natureza tecnológica, ambos constituintes da aplicação informática. A *UML* é um importante auxiliar para o domínio da complexidade das regras a cumprir e na definição de processos e fluxos de informação.

Assim, e de forma a auxiliar a fase de implementação, foi construído o diagrama de *use cases* (figura 6.17) que pretende representar o levantamento dos requisitos do sistema na perspectiva do utilizador, estando de acordo com as suas necessidades [23]. Pode-se verificar que existe um único actor neste processo, o utilizador, que poderá efectuar um conjunto de acções sobre o sistema, como seleccionar o projecto, validar dados introduzidos e consultar dos ficheiros produzidos pela aplicação.

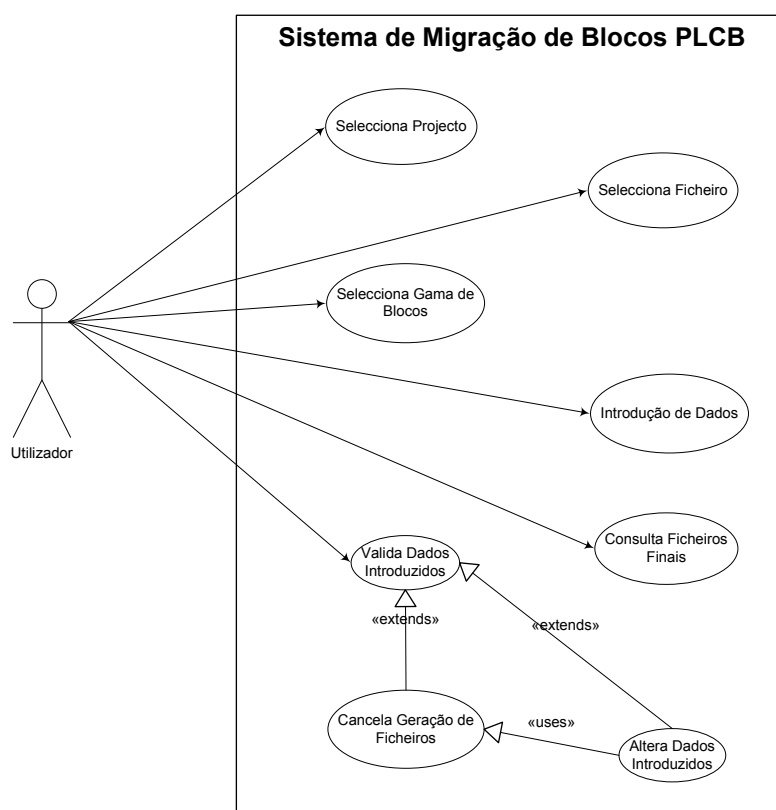


Figura 6.17 - Diagrama de *use cases* da aplicação.

O diagrama de classes da aplicação constante nas figuras 6.18 e 6.19. Com este diagrama pretende-se descrever formalmente a estrutura de objectos na aplicação e, para cada objecto, descrever a sua identidade, os seus relacionamentos, atributos e operações.

Devido à extensão do diagrama de classes optou-se pela sua divisão de forma a torná-los perceptíveis. A divisão foi efectuada a partir dos módulos com as suas características comuns, visto poderem ser de diferentes tipos.

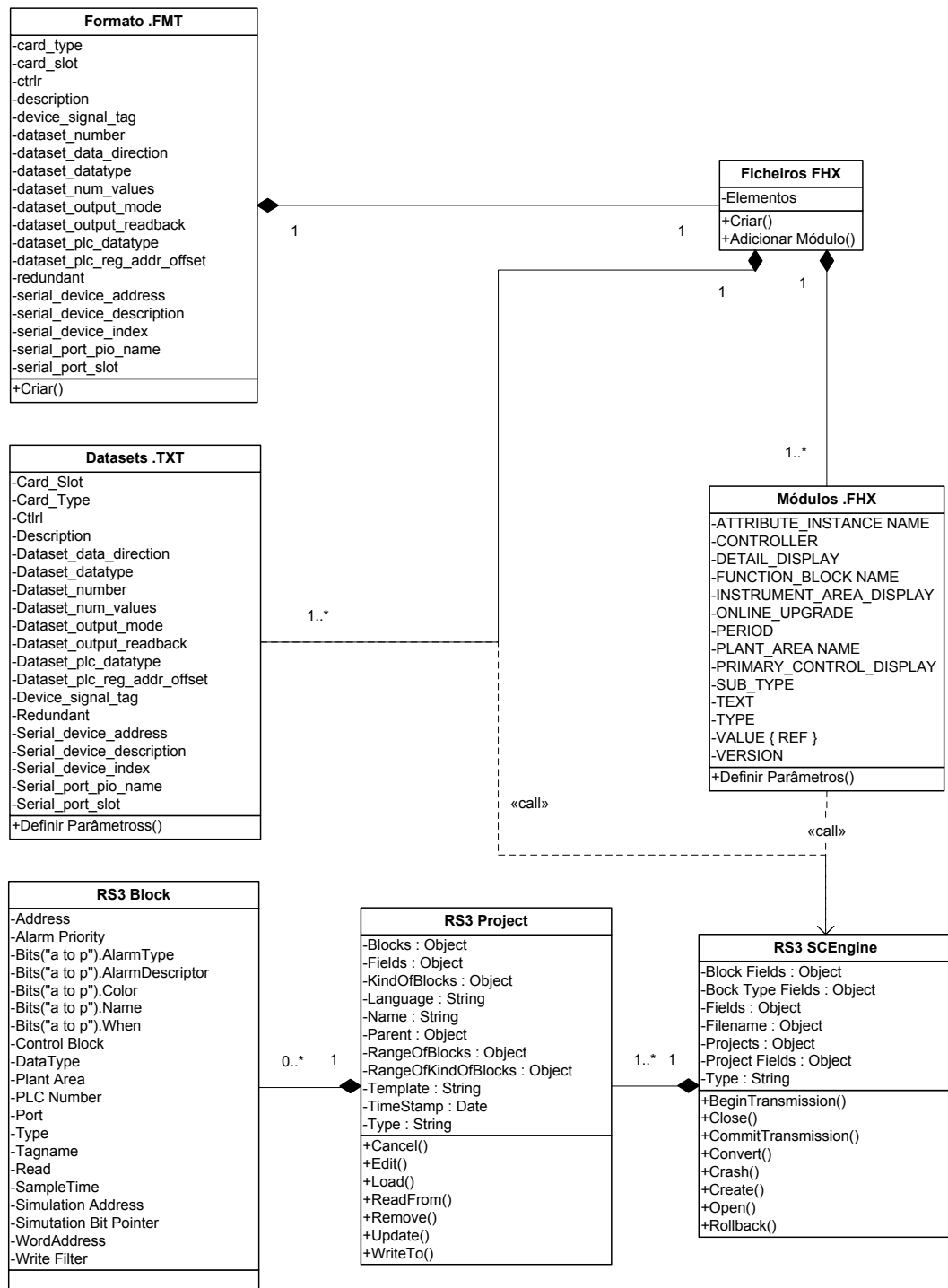


Figura 6.18 - Diagrama de classes da aplicação.

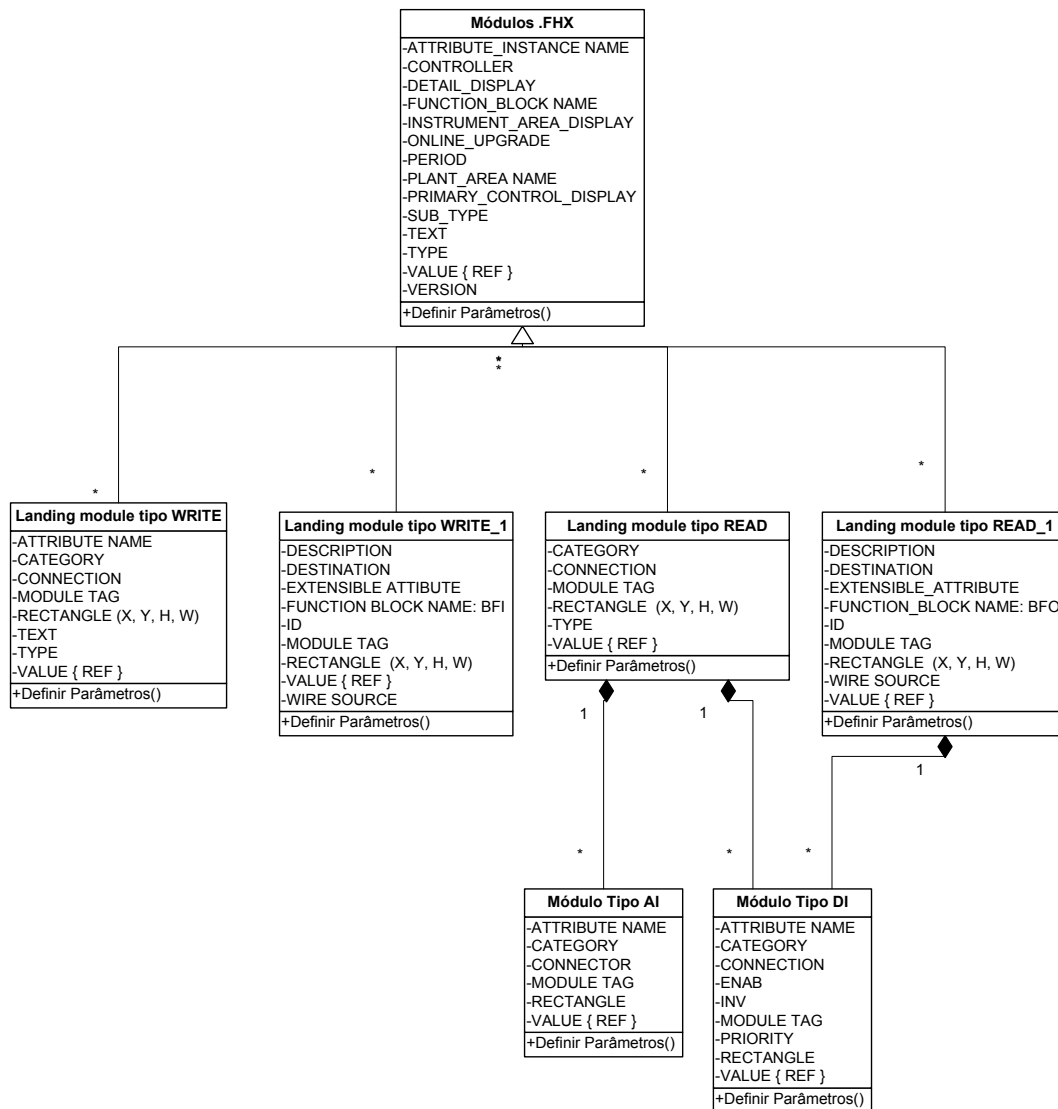


Figura 6.19 - Diagrama de classes da aplicação (continuação).

A figura 6.18 mostra que a aplicação tem por base um ficheiro de especificação de formato, um ficheiro relativo aos *datasets* e diversos ficheiros de configuração para criação de módulos.

Toda a informação necessária para o ficheiro de *dataset* e para os módulos têm como base a biblioteca de objectos *COM* do *CS for RS3*, sendo esta composta por várias bases de dados que por sua vez poderão conter inúmeros projectos.

Na figura 6.19, mostra os diferentes módulos criados pela aplicação: *landing modules*, que, consoante os registos são ou não consecutivos e baseado no tipo de bloco do *RS3* podem ser de quatro formas distintas: *read*, *write*, *read_1* e *write_1*, e os módulos individuais *AI* e *DI* para valores analógicos e discretos, respectivamente. Estes últimos são criados a partir de dados provenientes dos *landing modules* que os precederam e podem dar origem a inúmeros ficheiros.

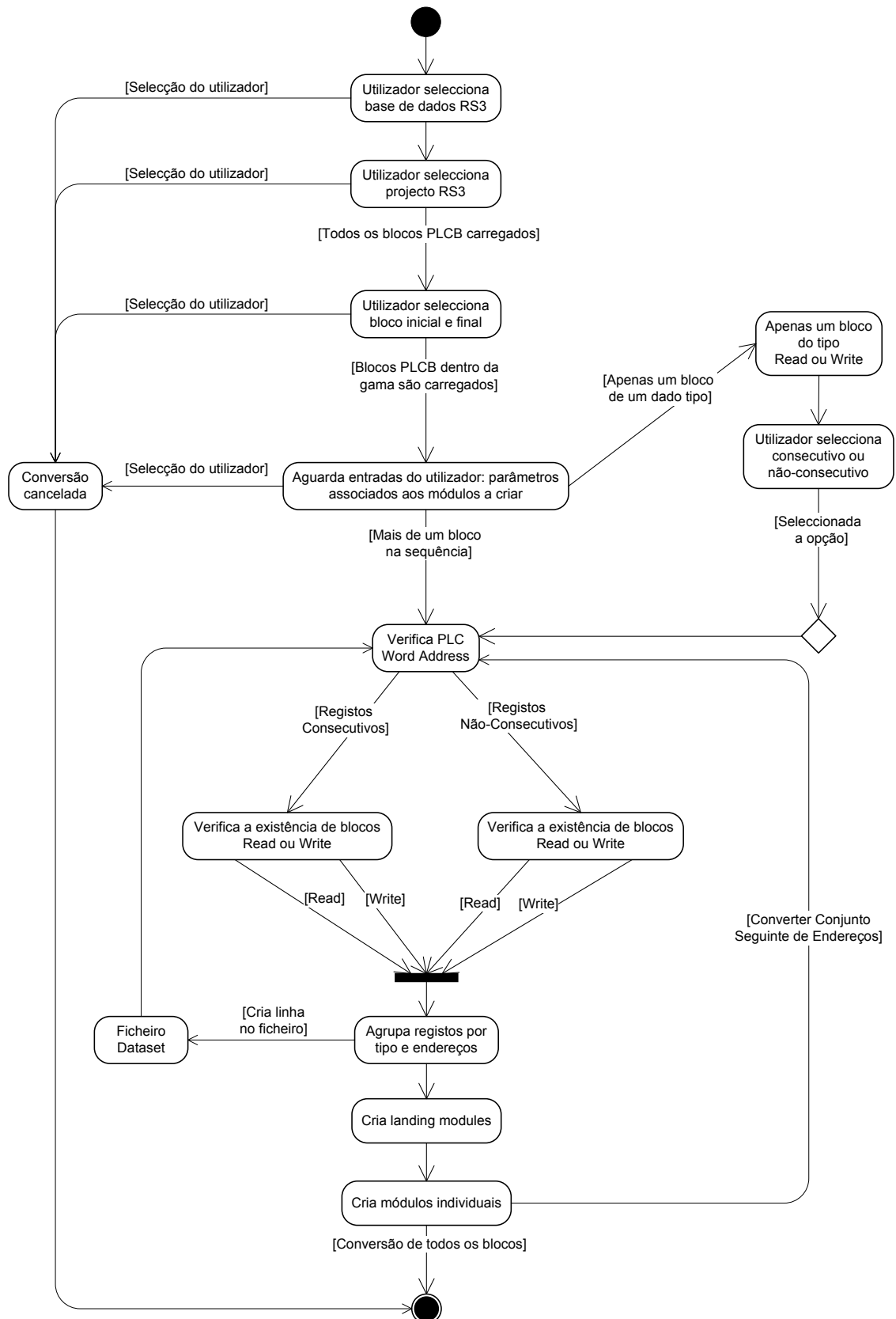


Figura 6.20 - Máquina de estados da aplicação.

Na figura 6.20 é apresentada a máquina de estados da aplicação que permite, de forma precisa, verificar como opera a ferramenta criada. Inicialmente o utilizador terá que seleccionar uma das bases de dados onde se encontram os blocos a converter. Em seguida deverá seleccionar o projecto, visto que em cada base de dados do *CS for RS3* poderão ser criados/importados inúmeros projectos. A partir deste ponto a aplicação deverá listar todos os blocos do tipo *PLCB*, existentes no projecto, por ordem de endereço de *RS3*, mas listando o *tag* e não o endereço.

O utilizador deverá, agora, seleccionar a gama de blocos *PLCB* que pretende converter, para que apenas estes sejam analisados pela aplicação. Tendo que introduzir mais alguns dados para que a aplicação possa prosseguir, o utilizador só terá que intervir uma vez mais no caso de serem encontrados blocos isolados de leitura ou escrita, visto que a aplicação com um só bloco não poderá determinar se os registos são ou não consecutivos. Assim sendo, deverá ser o utilizador a seleccionar para que a conversão dos blocos continue.

A selecção entre registos consecutivos e não consecutivos é de vital importância na determinação do passo seguinte, ou seja, no agrupamento dos registos por tipo, visto que por registo já o foram anteriormente. É agora possível obter o ficheiro *dataset* e os *landing modules*.

Terminada esta fase os módulos individuais *AI* e *DI* são criados com referência a dados gerados pela própria aplicação e por dados obtidos do *CS for RS3*. A conversão dos blocos é então terminada com sucesso.

6.1.2.3. Migração da Configuração de Válvulas e Bombas

Foi elaborado um diagrama da figura 6.21 com o princípio de funcionamento do módulo com o bloco de função *DC* e respectivas funcionalidades requeridas, de forma a ir ao encontro de todos os requisitos referidos anteriormente.

Na figura é possível distinguir dois sinais de vital importância para todo o funcionamento do módulo: a confirmação de arranque e a confirmação de paragem. No entanto, em muitos casos, não é possível ter os dois sinais disponíveis no sistema, pois implica que existam fisicamente dois sinais de confirmação, o que nem sempre acontece. Assim sendo, quando um dos sinais não existe será utilizada a negação do sinal existente como forma de obter tal confirmação, ou seja, se só existir o sinal de confirmação de arranque, o equipamento estará parado quando o sinal de arranque não estiver activo.

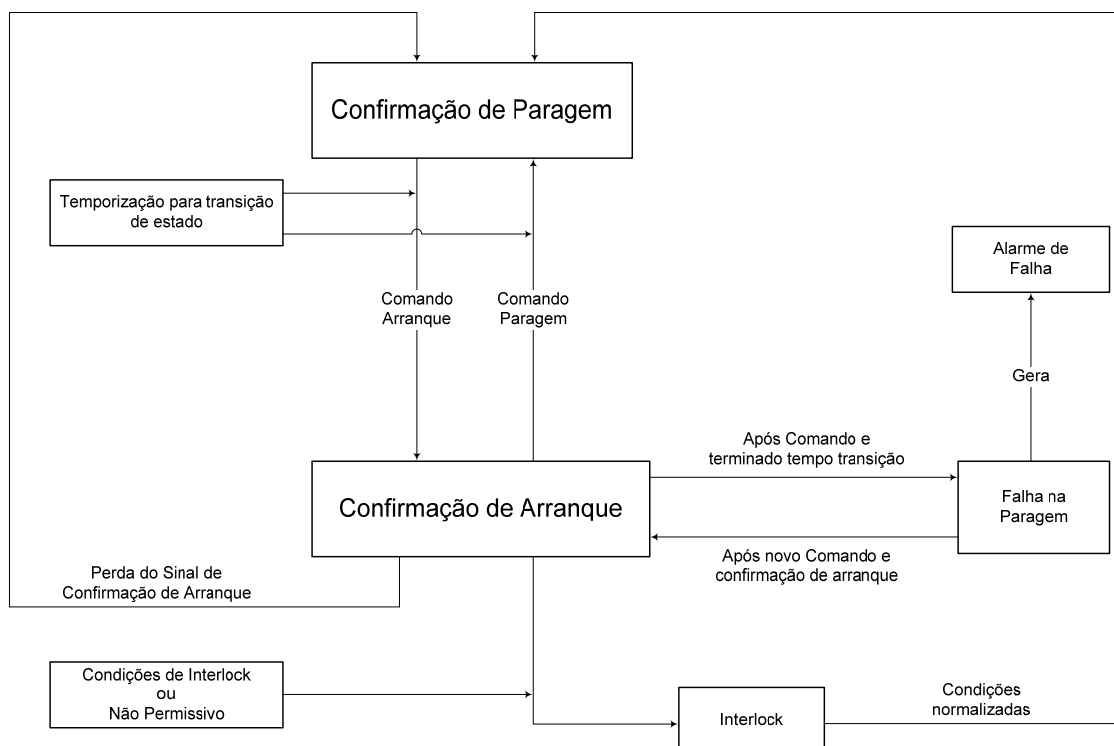


Figura 6.21 - Princípio de funcionamento do módulo para gestão de válvulas e bombas.

De forma a descrever todas as possíveis actuações do operador sobre o equipamento foi elaborada a figura 6.22. Os possíveis comandos serão dados a partir do *faceplate* relativo ao módulo *DC* do *DeltaV*.

Para além dos comandos mais habituais, arranque e paragem, pode existir também o sinal de *stop* (paragem do equipamento numa posição intermédia) que é utilizado para válvulas motorizadas. A existência deste sinal implica que tenha que existir um outro sinal de saída do bloco de função *DC* embora, ao nível de funcionalidade do próprio bloco, não implique alterações de fundo.

Como se pode verificar o utilizador perde actuação no equipamento sempre que haja um *interlock* activo quer seja devido a condições processuais ou do próprio equipamento (por exemplo sobreaquecimento). Após as condições estarem devidamente normalizadas o operador terá, novamente, acesso ao equipamento.

A execução de *bypass* a um dado *interlock* é possível sempre que ele seja processual. Caso esteja relacionado com aspectos físicos do equipamento não será dada a possibilidade de efectuar *bypass* de modo a não danificá-lo.

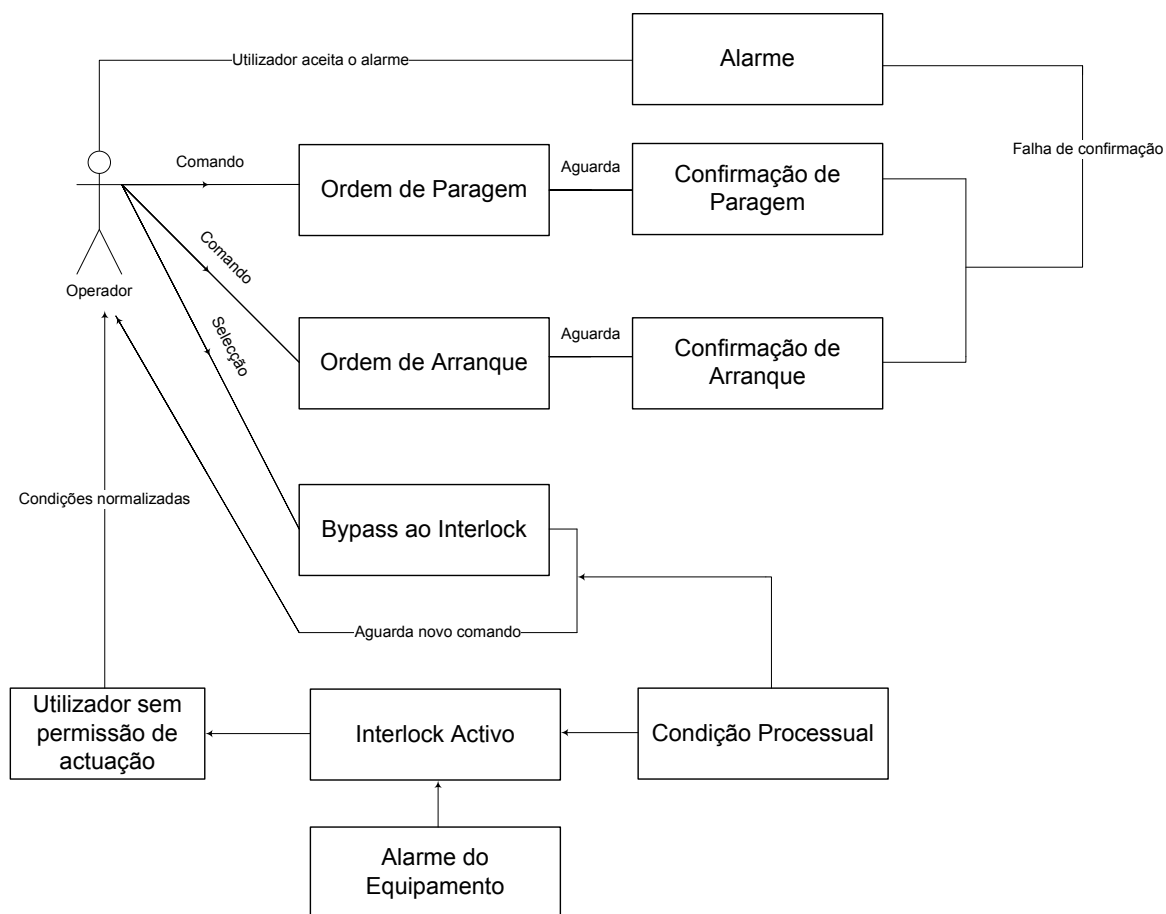


Figura 6.22 - Papel do operador na gestão de válvulas e bombas a partir do *faceplate*.

Capítulo 7

Implementação e Testes

Após a definição da arquitectura da aplicação a desenvolver para a migração desde o *RS3*, é agora o momento da sua implementação e teste. Neste capítulo é abordado a forma como foi desenvolvida a aplicação, a ferramenta base para o seu desenvolvimento, o seu funcionamento e os testes realizados de forma a validar a aplicação.

Em relação ao novo módulo para válvulas e bombas e respectivos *faceplates* serão, igualmente, desenvolvidos alguns pormenores da sua implementação.

7.1. Desenvolvimento da Aplicação

Baseada na análise de requisitos e nos diagramas em *UML* apresentados no capítulo 6, a aplicação de migração automatizada dos blocos *PLCB* desde o *RS3* para o *DeltaV* foi implementada recorrendo a três ferramentas:

- à linguagem de programação *Visual Basic* da *Microsoft*, que é parte integrante do pacote *Microsoft Visual Studio*,
- ao *CS for RS3*,
- ao *software DeltaV* essencialmente através de uma das suas aplicações, o *DeltaV Explorer*.

As duas últimas ferramentas foram desenvolvidas pela *Emerson Process Management*.

A utilização da linguagem de programação *Visual Basic* foi natural devido ao facto de algumas das ferramentas utilizadas actualmente na refinaria terem por base esta linguagem.

O *CS for RS3*, que foi largamente discutido no capítulo 5, é também uma das ferramentas essenciais utilizadas na implementação da aplicação por permitir a edição *offline* da base de dados do sistema *RS3* e, portanto, anular qualquer perturbação processual.

Através do *DeltaV Explorer* será possível efectuar a importação de todos os ficheiros gerados pela aplicação, utilizando a ferramenta *Standard DeltaV Import* ou *User DeltaV Import*. A primeira opção será usada quando forem obtidos ficheiros no formato *.FHX* e a segunda no formato *.TXT*, sendo então necessário o ficheiro de especificação de formato *.FMT*.

7.2. Interface da Aplicação com o Utilizador

A primeira fase de desenvolvimento da aplicação passou pela criação do ficheiro de especificação de formato com todos os parâmetros essenciais à correcta importação do ficheiro *.TXT* que será gerado.

Foram inicialmente criados três formatos diferentes e foram testados todos eles para verificar qual deles teria o número de parâmetros essencial para este projecto. A diferença entre os três ficheiros era apenas no número de parâmetros. Verificou-se que um deles tinha demasiados parâmetros e que alguns deles não eram importantes para a criação dos *datasets* e que outro continha um número muito reduzido de parâmetros e pecava pela inexistência de alguns dados importantes, embora não essenciais.

Após ter sido encontrado o ficheiro mais adequado foi necessário conhecer profundamente a ferramenta *CS for RS3* e a sua biblioteca de objectos *COM*. Foram efectuados testes recorrendo ao *Visual Basic* para aceder a alguns parâmetros dos blocos em estudo.

Em seguida passou-se à criação do primeiro ficheiro com extensão *TXT* para a criação de *datasets*, correspondente à fase I do trabalho.

Na figura 7.1 apresenta-se a primeira janela da aplicação onde será seleccionada a base de dados pretendida no formato *CSC*. Na janela seguinte será solicitado ao utilizador que seleccione dentro da base de dados o projecto onde se encontram os blocos para migração (figura 7.2).



Figura 7.1 - Primeira janela da aplicação para selecção da base de dados do CS for RS3.

Por defeito quando se prime a tecla *Avançar* na primeira janela, a janela seguinte tem seleccionado o primeiro projecto por ordem alfabética e são listados nos campos *TAG Inicial* e *TAG Final* o nome do primeiro *tag* encontrado do tipo *PLCB*. A alteração do projecto implica automaticamente a alteração do *TAG Inicial* e *TAG Final* pois é executada nova busca no projecto para encontrar novamente o primeiro bloco *PLCB*.

Projecto:

Nome do ficheiro: Maximo de 6 caracteres

1. Device_Signal_Tag

Device_Signal_Tag:

TAG Inicial: TAG Final:

2. Descrição:

3. Ctrl:

4. Card_Slot:

5. Card_Type: 6. Redundant:

7. Serial_Port_Slot: 8. Serial_port_pio_name:

9. Serial_device_index: 10. Serial_device_description:

11. Serial Device Address: 12. Dataset_number:

13. Dataset_data_direction: 14. Dataset_type:

15. Dataset_num_values: 16. Dataset_output_mode:

17. Dataset_output_readback: 18. Dataset_plc_datatype:

19. Dataset_plc_reg_addr_offset: 20. Primary_Control_Display:

Gerar Fechar

Figura 7.2 - Segunda janela para selecção do projecto e configuração de parâmetros adicionais para criação de *datasets* e módulos.

Em seguida o utilizador poderá preencher os campos assinalados a cor-de-rosa. Estes campos são opcionais, ou seja, o ficheiro criado não terá qualquer problema ao nível de importação para o *DeltaV*, no entanto, o seu não preenchimento conduz a ficheiros com pouco informação, visto que dois dos parâmetros são meramente descritivos: *Descrição* e *Serial_Device_Description*.

O campo do sinóptico *Primary_Control_Display* sendo igualmente opcional permite, em caso de alarme conduzir o operador directamente ao sinóptico onde se encontram apresentados os valores processuais, o que torna a sua configuração importante mas não essencial. Esta funcionalidade não estava disponível no *RS3*. O operador, em caso de alarme, teria que se recordar o sinóptico exacto onde este aparece de forma a analisar as potenciais causas da anomalia.

Na figura 7.3 é apresentada a mesma janela da aplicação com os parâmetros já preenchidos: foi seleccionado o projecto *PTG4-070*, cujo primeiro *tag* *PLCB* pretendido é o *FGFOB-01* e o último é *FG29FOB-13*. O controlador onde será colocado o *dataset* é o *CTLR-FOB-3* na carta 45 na porta 1. O sinóptico a abrir em caso de alarme é o *S2301*.

Os parâmetros assinalados na segunda metade da janela da figura 7.3 permitem acompanhar a criação das linhas, referenciado o tipo de dado que se encontra a converter e

parâmetros que são colocados por defeito. Por exemplo o parâmetro *Card_Type* será *SERIAL_32DS_SER2*, referindo que se trata de uma carta série de *DeltaV* com um máximo de 32 *datasets* apresentando a 2.^a versão de *firmware*.

The screenshot shows a window titled "Criação de DataSets" with the following fields and values:

- Projecto:** PTG4-070
- Nome do ficheiro:** TESTE (Maximo de 6 caracteres)
- 1. Device_Signal_Tag:**
 - Device_Signal_Tag: FG
 - TAG Inicial: FGF0B-01
 - TAG Final: FG29F0B-13
- 2. Descrição:** Teste Fase I
- 3. Ctrl:** CTRL-F0B-3
- 4. Card_Slot:** 45
- 5. Card_Type:** SERIAL_32DS_SER2
- 6. Redundant:** F
- 7. Serial_Port_Slot:** 1
- 8. Serial_port_pio_name:** SERIAL_PORT
- 9. Serial_device_index:** 1
- 10. Serial_device_description:** Teste Fase 1
- 11. Serial Device Address:** 1
- 12. Dataset_number:** 0
- 13. Dataset_data_direction:** (empty)
- 14. Dataset_type:** (empty)
- 15. Dataset_num_values:** (empty)
- 16. Dataset_output_mode:** complete block
- 17. Dataset_output_readback:** F
- 18. Dataset_plc_datatype:** (empty)
- 19. Dataset_plc_reg_addr_offset:** (empty)
- 20. Primary_Control_Display:** S2301

Buttons at the bottom: Gerar, Fechar.

Figura 7.3 - Segunda janela para selecção do projecto e configuração de parâmetros adicionais para criação de *datasets* e módulos.

Premindo o botão Gerar surgirá uma nova janela, figura 7.4, que informa o utilizador que a conversão dos dados foi efectuada.

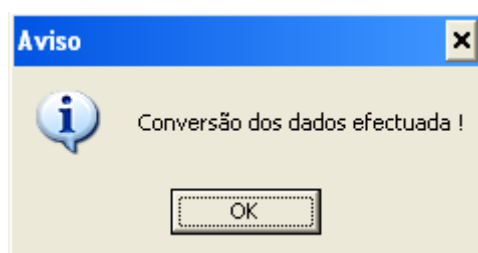


Figura 7.3 - Janela informativa de finalização da criação de *datasets*.

Terminada a primeira fase do trabalho e respectiva validação foi possível transitar para a fase II. Nesta fase de criação de ficheiros FHX o utilizador não necessitará de introduzir mais nenhum parâmetro.

Durante a análise do tipo de ficheiros e da sequência de registo surge uma situação que, embora não muito usual, é possível aparecer: a existência de um único bloco de leitura ou escrita. Neste caso a aplicação não consegue determinar sozinha se este bloco deverá ser tratado como um conjunto de registos consecutivos ou não. Assim sendo, terá que ser o utilizador a saber. A janela da figura 7.4 é colocada ao utilizador para selecção.

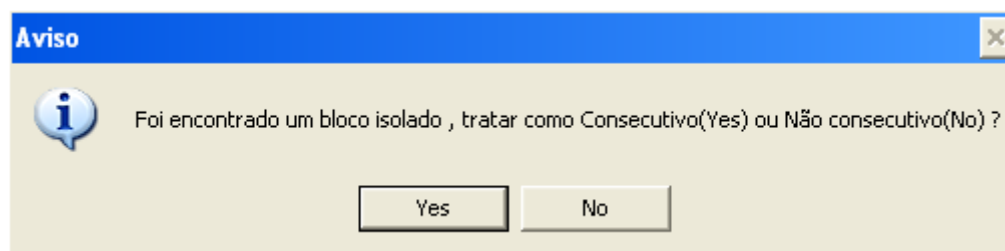


Figura 7.4 - Janela para selecção de registos consecutivos ou não quando existe apenas um bloco de leitura ou escrita.

Assim que o utilizador selecciona a criação de *datasets* e ficheiros é iniciada. Para que o utilizador esteja a par do processamento foram criados dois parâmetros que vão alterando consoante está a criar um *dataset* ou módulos. Na figura 7.5 é apresentada a janela correspondente à criação de um *dataset* e na figura 7.6 a correspondente à criação de módulos, onde consta sequencialmente o registo inicial, o número de registos e qual o registo em processamento.

17. Dataset_output_readback	F	18. Dataset_plc_datatype
19. Dataset_plc_reg_addr_offset	0	20. Primary_Control_Display

À criar o ficheiro...

Figura 7.5 - Pormenor do parâmetro informativo da criação de *dataset*.

17. Dataset_output_readback	F	18. Dataset_plc_datatype
19. Dataset_plc_reg_addr_offset	5128	20. Primary_Control_Display

A processar 5193 / 14 - 14

Figura 7.6 - Pormenor do parâmetro informativo da criação de módulos onde consta o número do registo, o número de registos a processar e o registo actual em processamento.

Quando termina a conversão do conjunto de registos com a criação do ficheiro *TXT* e dos ficheiros *FHX* surgirá a janela da figura 7.3, à imagem do que acontecia na fase I. Todos os ficheiros são guardados numa pasta denominada por *docs* (figura 7.7).

Name	Size	Type
FGFOB_SYS_1.fhx	4 KB	FHX File
FGFOB_SYS_2.fhx	4 KB	FHX File
FGFOB_SYS_3.fhx	4 KB	FHX File
FGFOB_SYS_4.fhx	4 KB	FHX File
FGFOB_SYS_5.fhx	4 KB	FHX File
FGFOB_SYS_6.fhx	4 KB	FHX File
FGFOB_SYS_7.fhx	4 KB	FHX File
FGFOB_SYS_8.fhx	4 KB	FHX File
FGFOB_SYS_9.fhx	4 KB	FHX File
FGFOB_SYS_10.fhx	4 KB	FHX File
FGFOB_SYS_11.fhx	4 KB	FHX File
FGFOB_SYS_12.fhx	4 KB	FHX File
FOB_AG.fhx	4 KB	FHX File
LS2101.fhx	4 KB	FHX File
LUB_AG.fhx	4 KB	FHX File
MPT_AG.fhx	4 KB	FHX File
PB2101_PB2102.fhx	4 KB	FHX File
PB2105_PB2104.fhx	4 KB	FHX File
PB2107_PB2108.fhx	4 KB	FHX File
PFC_AG.fhx	4 KB	FHX File
PFU_AG.fhx	4 KB	FHX File
PSH2123.fhx	4 KB	FHX File
PSH2124.fhx	4 KB	FHX File
SOV2106_07_RS3.fhx	4 KB	FHX File
SOV2108_09_RS3.fhx	4 KB	FHX File
SOV2110_RS3.fhx	4 KB	FHX File
TESTE.txt	1 KB	Text Document
TESTE_1_PLC_READ.fhx	15 KB	FHX File
TESTE_2_PLC_READ.fhx	3 KB	FHX File
TESTE_3_PLC_READ.fhx	15 KB	FHX File
TESTE_4_PLC_READ.fhx	6 KB	FHX File
TESTE_PLC_READ.fhx	15 KB	FHX File
TESTE_PLC_WRITE.fhx	4 KB	FHX File

Figura 7.7 - Conteúdo parcial da pasta *docs* onde são colocados todos os ficheiros *.TXT* e *.FHX*.

7.3. Implementação dos Módulos para Válvulas e Bombas

Na implementação dos módulos para válvulas e bombas e respectivos *faceplates* no *DeltaV* foram utilizadas duas ferramentas: o *DeltaV Operate* e o *DeltaV Explorer*. O primeiro permite a configuração da interface de operação e foi utilizado para o desenvolvimento dos *faceplates* e o segundo para a criação do módulo recorrendo à biblioteca de blocos de função.

7.3.1. Desenvolvimento dos Módulos e *Faceplates*

Através do princípio de funcionamento da figura 6.20, o novo módulo foi configurado com dez blocos de condição que têm como objectivo definir condições que, quando activas, o

operador não tenha permissão para operar o equipamento. Os blocos de condições são ligados a um bloco *BFI* cuja função é verificar se existe alguma condição activa e posteriormente passar essa informação ao bloco *DC*. Para ser possível efectuar um *bypass* às condições utilizou-se o bloco *OR* ligado ao parâmetro *Disable* de cada condição, permitindo activar ou desactivar cada condição individualmente.

Caso uma ou mais condições estejam activas, surgirá no *faceplate* (figura 7.9) um sinal de *Permit* com um círculo vermelho para avisar o operador que não tem permissão para operar. Neste *faceplate* foi configurado um maior número de botões relativamente ao de base: abertura do sinóptico onde se encontra o equipamento, acesso ao *CS* do *DeltaV* da figura 7.8 e um botão que permite aceitar todos os alarmes activos daquele módulo.

O *faceplate* apresentado é referente a uma bomba. Para o caso de uma válvula o *faceplate* é semelhante mas em vez de comando de *start/stop* aparecerá *open/close*, ou seja, são alteradas as denominações para efectuar o enquadramento do tipo de equipamento a operar.

De forma a saber qual a condição activa em cada momento que não permite o acesso ao equipamento desde o *DCS* é indicado no *faceplate* de detalhe (figura 7.10) um descritivo dessa condição e uma seta vermelha apontando para a(s) referida(s) condição(ões), com a possibilidade de efectuar o respectivo *bypass*. Foi igualmente adicionada uma área para visualização dos diagnósticos do módulo e respectivo alarmes de falha de equipamento. Este sinal é físico ou proveniente de comunicação *MODBUS*. Em ambos os casos deverá ser configurado o sinal na carta pretendido e colocada a referência correspondente neste módulo. Caso não exista sinal de falha basta colocar a opção *Enable* em *False* directamente no módulo do equipamento (figura 7.8).

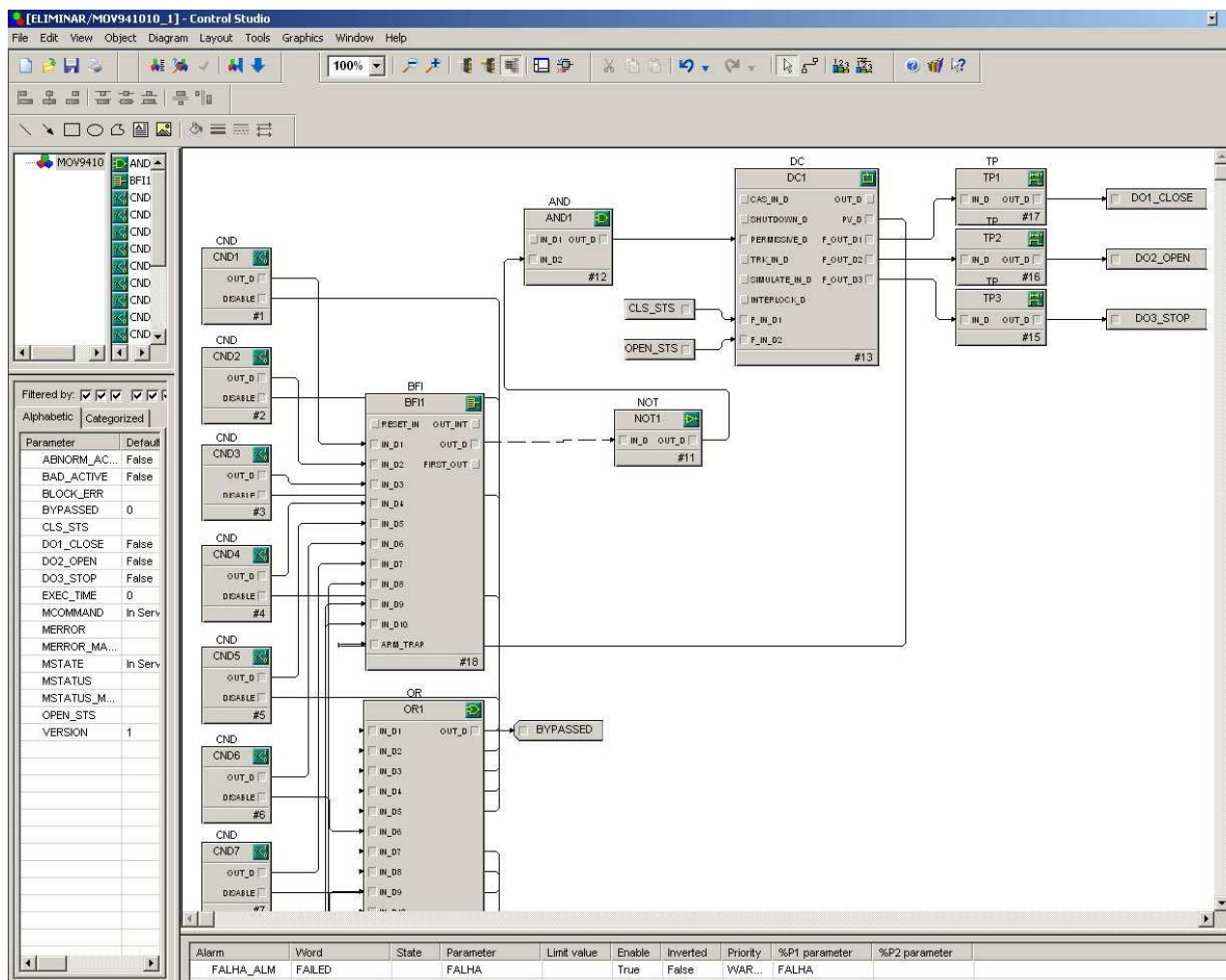


Figura 7.8 - Módulo para operação de válvulas e bombas no DeltaV.

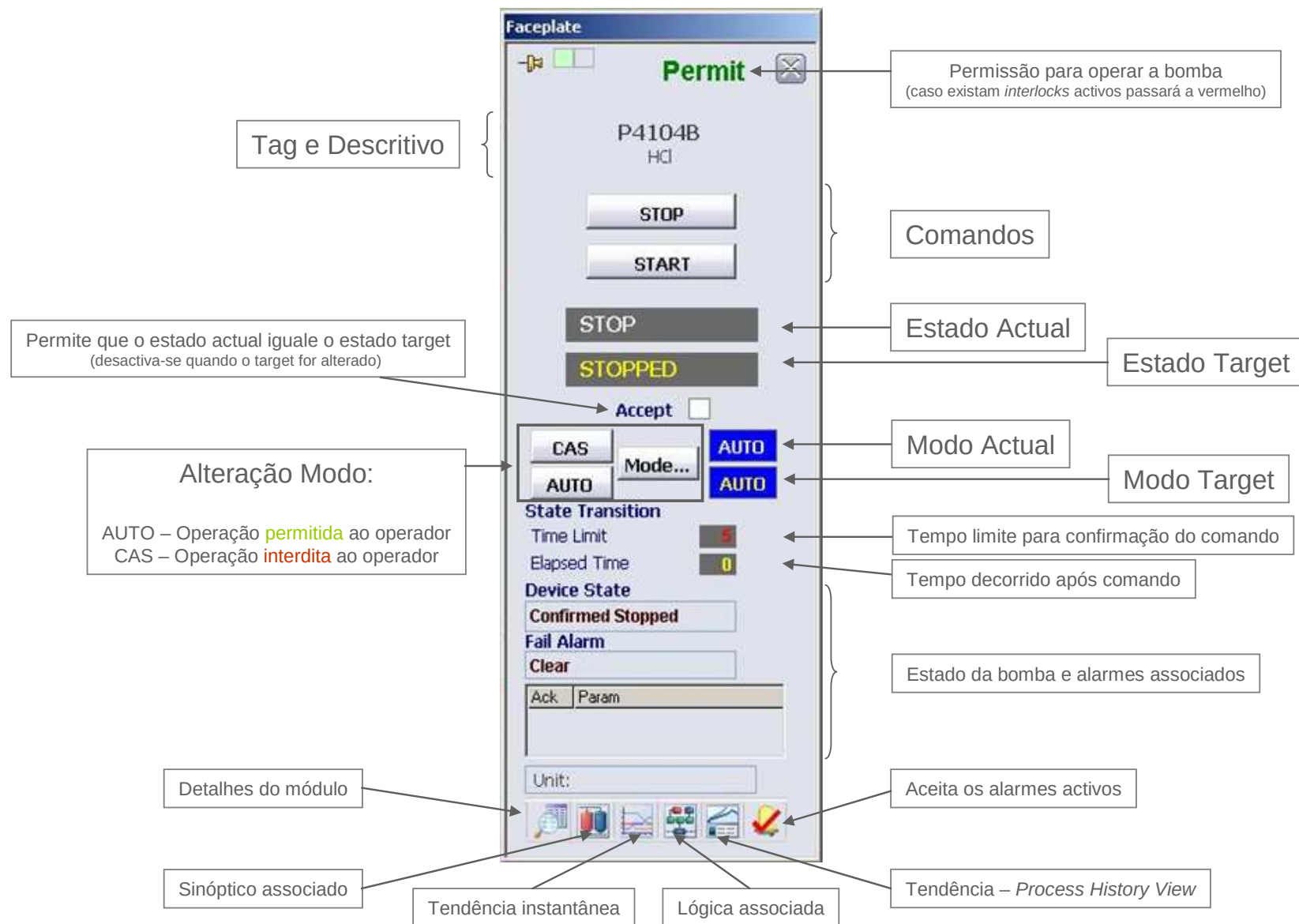


Figura 7.9 - Faceplate para operação de bomba no DeltaV com explicação das funcionalidades.

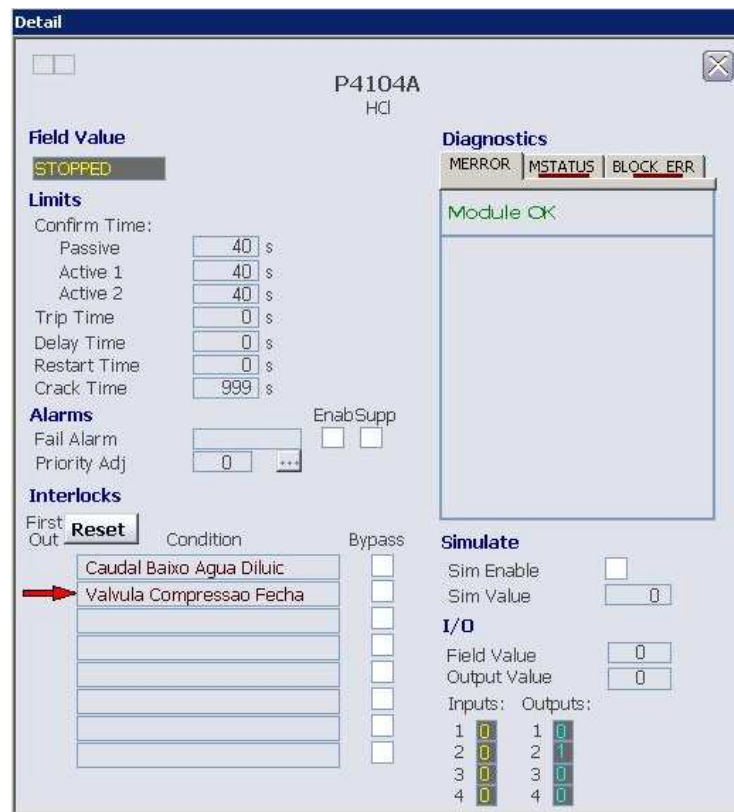


Figura 7.10 - Faceplate de detalhe para operação de bomba no DeltaV.

Como foi anteriormente referido a configuração de ambos os *faceplates* foi efectuada na aplicação *DeltaV Operate*, utilizando funcionalidades embebidas, nomeadamente as caixas de diálogo referentes às animações pretendidas nos diversos parâmetros e botões.

A caixa de diálogo apresentada na figura 7.11 permite a animação de cor de *foreground*, *background* e *edge*, configurar comandos através de um *click* (por exemplo quando carrega num dado botão abre uma nova janela), em termos de movimento por alteração da posição, escala ou efeito de rotação (no caso de objectos). Pode ser utilizada a opção de *fill percentage* para animação do preenchimento de um objecto consoante o valor de uma variável (muito utilizado para animação de tanques com o valor do nível respectivo), ou afectar a própria visibilidade do parâmetro ou objecto.

Animações adicionais são possíveis em casos específicos mas não foram utilizados no âmbito deste trabalho.

Em alguns casos foi necessário recorrer ao código base em linguagem *Visual Basic* para efectuar algumas alterações e *debug* na colocação em funcionamento de ambos os *faceplates*.

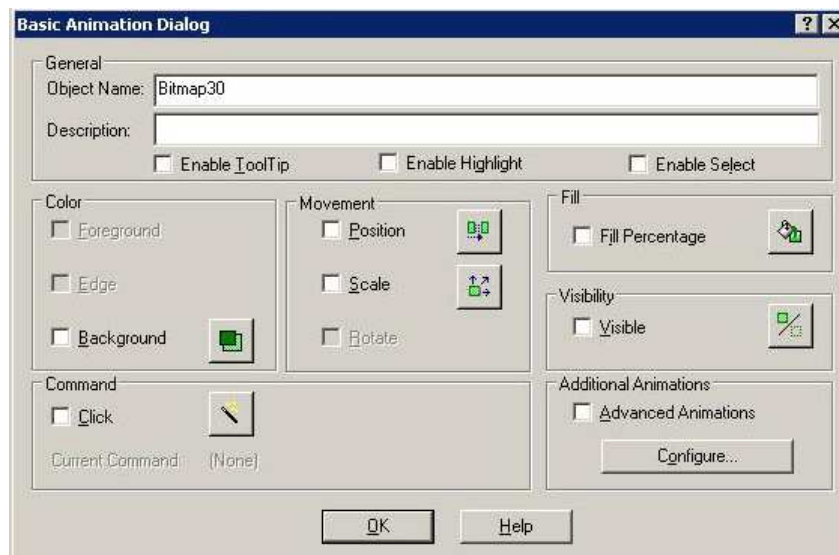


Figura 7.11 - Caixa de diálogo referente à animação de parâmetros e objectos no *DeltaV Operate*.

O bloco *DC* propriamente dito é composto por dois sinais de entrada e três sinais de saída (figura 7.12). Os sinais de entrada correspondem à confirmação de arranque/abertura e confirmação de paragem/fecho. Os sinais de saída correspondem aos comandos que são possíveis desde do sistema: arrancar/abrir, parar/fechar e o sinal de stop, que poderão ser comandos por pulso utilizando-se o bloco de função *TP* (*timed pulse*) ou por comando mantido em que será necessário eliminar os blocos *TP*.

Um dado equipamento poderá não suportar os três comandos e portanto será necessário eliminar os não utilizados da configuração conjuntamente com o bloco *TP* correspondente.

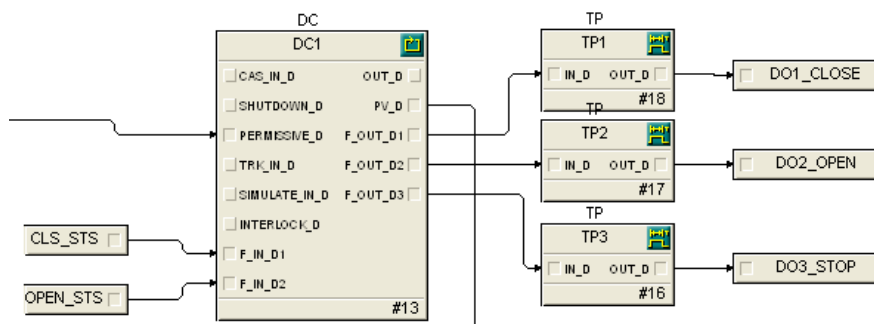


Figura 7.12 - Pormenor do bloco *DC* no módulo de *DeltaV*.

O bloco *DC* permite, como o seu semelhante no *RS3*, a configuração de diversas opções (figura 7.13). Estas opções permitem a definição da utilização ou não de *interlocks* e permissivos (baseado no conjunto de condições discutido atrás). Caso haja discordância entre o comando e o estado actual do equipamento durante um intervalo de tempo superior ao configurado será necessário efectuar *reset* no caso desta opção estar seleccionada.

A opção *SP Track* está relacionada com o seguimento de alterações de campo por parte do bloco *DC* quando o equipamento é retirado de comando à distância, passando a ser comandado apenas localmente.

A opção *Passive when Confirmed* efectua a alteração do comando para o estado passivo (que tanto poderá ser arranque ou paragem) quando o bloco *DC* recebe a confirmação do estado do equipamento. A opção *Passive on Active Timeout* é utilizada quando foi atingido o intervalo de tempo configurado para a recepção da confirmação do estado do equipamento.

O bloco *DC* suporta apenas dois modos: automático que corresponde ao manual no sistema *RS3* e *cascade* que corresponde ao automático no *RS3*, ou seja, permite a operação baseada em condições para arranque/paragem.

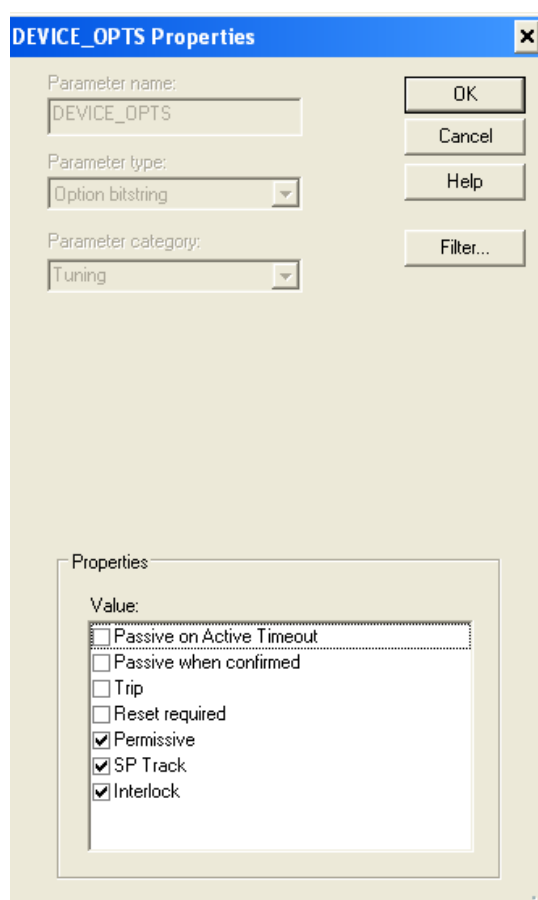


Figura 7.13 - Opções do DC no DeltaV.

7.3. Testes da Aplicação

Após a conclusão da implementação da aplicação vários testes foram levados a cabo de forma a validar o seu funcionamento.

7.3.1. Testes Preliminares

Uma vez terminados os testes de criação de ficheiros foi necessário proceder à validação de cada tipo de ficheiro criado *dataset*, *READ*, *READ_1*, *WRITE*, *WRITE_1*, *AI* e *DI*. Para tal exportaram-se alguns ficheiros em actual funcionamento da base de dados do *DeltaV* e efectuou-se a comparação utilizando uma folha de cálculo para verificar se o formato genérico correspondia e se eram alterados os parâmetros pretendidos.

Foram executados testes utilizando dados de *PLC* em actual funcionamento no *RS3*, através da selecção adequado do número de blocos e, consequentemente, do número de registos. Foram contabilizadas manualmente o número de registos, o número de linhas no ficheiro *datasets* e o número de ficheiros *.FHX* que deveriam ser gerados pela aplicação. Quando esta etapa foi concluída passou-se à validação do conteúdo dos ficheiros de forma a permitir uma importação sem erros no *DeltaV*. O conteúdo foi, novamente, comparado recorrendo a diversas folhas de cálculo em que a não coincidência das linhas foi analisada ao pormenor.

Terminada mais uma fase de testes foram iniciadas as importações dos diversos ficheiros para o *DeltaV* utilizando o *DeltaV Explorer*. Foram corrigidos erros de importação, particularmente relacionados com os diversos parâmetros de data e hora existentes nos ficheiros *.FHX*, que implicam a não importação dos mesmos no caso do formato não ser adequado. Visto terem sido efectuadas várias importações do mesmo conjunto de ficheiros e, visto que a substituição dos módulos é imediata e sem aviso prévio, implica que a importação para o sistema real tenha que ser efectuada cuidadosamente sob pena de alterar módulos em funcionamento, em caso de eventual erro da aplicação.

Para minimizar eventuais erros deste tipo, e para que todos os testes ocorressem num ambiente seguro, foi instalado num *PC* o *software DeltaV* e o *CS for RS3* e importada a actual base de dados utilizada na refinaria, permitindo efectuar todos os testes preliminares sem afectar a base de dados em funcionamento.

Foram igualmente testados encravamentos, recuperação em caso de erro e despiste de eventuais situações anómalas ou de rara ocorrência.

No anexo 8 estão listados alguns dos testes preliminares que foram efectuados para a fase I e no anexo 9 os testes executados para a fase II.

7.3.2. Testes de Validação

O primeiro ficheiro a ser importado para o *DeltaV* foi o *dataset* utilizando o formato *Serial_Dataset.fmt* previamente criado. Na figura 7.14 é possível verificar que a importação foi efectuada com sucesso.

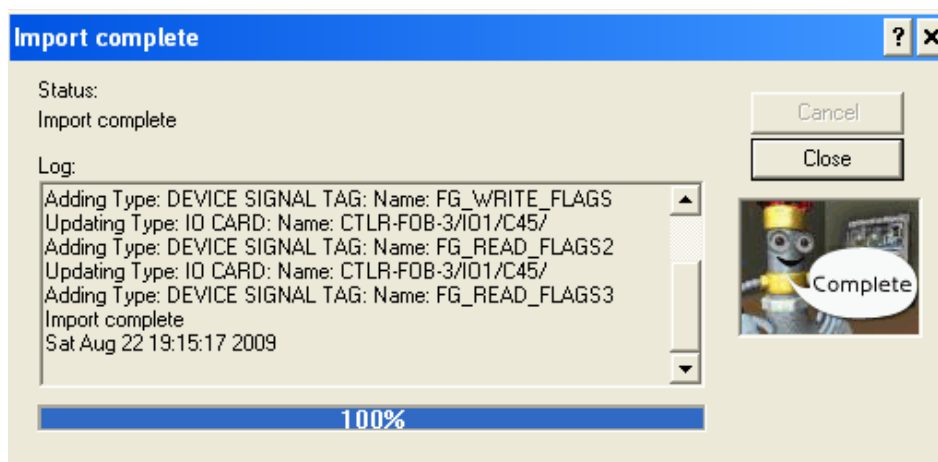


Figura 7.14 - Resultado da importação do ficheiro *datasets* para o *DeltaV*.

No *DeltaV Explorer* verificou-se a criação dos *datasets* no controlador, carta e porta seleccionadas pelo utilizador (figura 7.15). Dentro de cada *dataset* foi verificada a existência do conjunto de registos pretendido bem como as características de cada um.

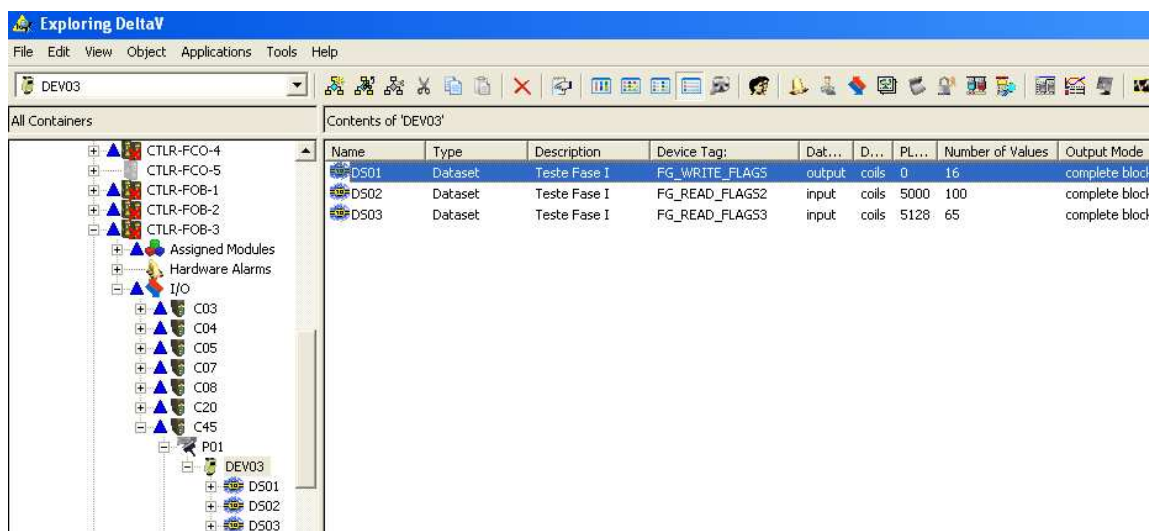


Figura 7.15 - Resultado da importação do ficheiro *datasets* para o *DeltaV*.

Para executar uma validação mais profunda foram utilizados dados de dois *PLC* mas que já se encontram migrados para o *DeltaV*. Comparou-se o número de ficheiros e o seu conteúdo especialmente o registo e eventuais alarmes. Efectuou-se a importação destes para a base de

dados que não estava em funcionamento e validou-se a utilização destes módulos, em especial os tipos *AI* e *DI*. A importação originou janelas semelhantes à figura 7.14.

Com a permissão da fábrica afectada foram eliminados dois módulos *AI* e dois módulos *DI* da base de dados em funcionamento e importados os quatro módulos respectivos que foram gerados pela aplicação. Foi necessário efectuar o *download* dos novos módulos de forma a serem reconhecidos pelo controlador. Confirmaram-se as indicações dadas pelos módulos, bem como a animação no sinóptico correspondente e foi solicitado um *force* de alarme no próprio *PLC* para um dos módulos *DI* de forma a testar a condição para alarme. No *DeltaV Operate* o operador verificou que o alarme apareceu na barra de alarmes e efectuou o reconhecimento do mesmo sem qualquer problema.

7.4. Testes dos Módulos para Válvulas e Bombas

Após a conclusão da criação do módulo e *faceplates* foram efectuados testes preliminares e de validação de forma a colocá-los em funcionamento de forma segura.

7.4.1. Testes Preliminares

Recorrendo ao *CS* do *DeltaV* em modo de simulação foi possível verificar e validar o funcionamento do módulo relativamente a comandos, encravamentos, accionamento do alarme de falha e *interlocks*.

Em seguida foram criados módulos separados para o sinal de falha, confirmação de arranque, confirmação de paragem. Em seguida configurou-se o módulo *DC* de forma a ler os valores destes novos módulos. Foram também criados os dois módulos de comando de arranque, paragem que recebem o sinal do módulo *DC*. Efectuaram-se novos testes para verificar se os comandos eram enviados para os módulos com os sinais de arranque e paragem e se alterações impostas nos sinais de falha e confirmações eram devidamente enviados ao módulo *DC*.

Todo este processo foi sendo acompanhado no *faceplate* e *faceplate* de detalhe onde foram confirmadas as alterações no estado do equipamento, alarmes e *interlocks*.

7.4.2. Testes de Validação

O módulo *DC* e os *faceplates* correspondentes foram colocados em funcionamento em Maio deste ano para duas bombas da *FUT*. Sucederam-se outras implementações na mesma fábrica mas em unidades processuais diversas, não tendo sido reportada qualquer anomalia.

Capítulo 8

Conclusões

O desenvolvimento da aplicação *MG_PLC* para migração automatizada das configurações desde o *DCS RS3* para o *DCS DeltaV* permite uma poupança de tempo muito significativa, visto que permite efectuá-la com um pequeno esforço de introdução de alguns parâmetros na aplicação, sendo o restante processo automático.

Em termos de métricas foi determinado que para um *PLC* com cerca de 300 registos são necessários cinco dias numa base de 8 horas de trabalho diárias para um configurador. Utilizando a ferramenta desenvolvida a configuração pode estar no sistema em cerca de 10 minutos, incluindo já o tempo necessário para a importação dos módulos, dependente apenas do *DeltaV*. O que significa que em 10 minutos se pode ter toda a configuração dos registos necessários, bem como todos os módulos de leitura/escrita nesses registos. O último passo a ser executado no *DeltaV* será o *download* da carta e módulos importados de modo a activá-los no sistema.

Além desta poupança temporal, a aplicação permite a eliminação de erros que ocorriam de forma frequente nestas migrações com implicações processuais muitas vezes significativas, ao nível de trocas de registos ou não configuração de importantes alarmes.

Em relação ao módulo *DC* para gestão de válvulas e bombas as vantagens prendem-se não tanto com a diminuição do tempo de configuração, mas sim com a alteração da forma de funcionamento, pois o módulo *DMC* do *RS3* é, na sua génese, ligeiramente diferente do seu semelhante no *DeltaV*. A informação disponibilizada ao operador é, agora, muito maior, permitindo saber o estado do equipamento bem como eventuais encravamentos que não permitam a operação do mesmo.

A nível pessoal e profissional, a execução deste projecto aumentou muito significativamente o meu conhecimento em termos de programação em *Visual Basic*, visto que os meus conhecimentos eram limitados no início do projecto. O domínio desta ferramenta bem como o aumento de conhecimentos em relação ao funcionamento do *DeltaV* permite encarar de forma mais tranquila eventuais futuros projectos dos quais venha a fazer parte.

Anexo 1

Blocos função do sistema RS3

Tabela 4.1 - Blocos de entrada e saída do sistema RS3.

Tipo	Descrição
Bloco de entrada analógica (AIB)	Recebe um sinal analógico de um equipamento de campo através de uma carta de interface de campo (FIC) e converte os dados para que possam ser utilizados pelo resto do sistema. Para além disso permite endereçar e condicionar sinais de entrada, gerar alarmes e traçar gráficos de tendência.
Bloco de saída analógica (AOB)	Prepara um sinal analógico para ser enviado para um equipamento de campo. Permite endereçamento e armazenamento dos sinais de saída, bem como escolher o tipo de acção (directa ou inversa, mediante o tipo de válvula).
Bloco de entrada/saída de impulsos (PIOB)	Este bloco recebe um impulso do campo e gera impulsos de saída com uma duração e frequência específicas, para além disso possibilita fazer endereçamento e condicionamento do sinal de entrada, bem como armazenamento e endereçamento de sinais de saída.
Bloco de entrada de temperatura (TIB)	Recebe uma medição de temperatura proveniente de um termopar ou termistor (RTD) e converte o sinal para que possa ser utilizado pelo resto do sistema.
Bloco de entrada/saída redundante (RIOB)	Realiza uma cópia de segurança da FIC, monitorizando a sua “saúde” e estado da cópia de segurança.
Bloco de entrada de contacto (CIB)	Recebe um sinal discreto através de um opto acoplador e de uma FIC e prepara o sinal para que possa ser utilizado pelo resto do sistema. Permite ainda condicionar e endereçar o sinal de entrada, gerar alarmes, registar eventos e traçar gráficos de tendência.
Bloco de saída de contacto (COB)	Converte o estado de uma saída discreta para que esta possa ser enviada para o campo. Possibilita fazer armazenamento e endereçamento de sinais de saída discretos, bem como gerar uma saída invertida relativamente à entrada.
Bloco de entradas multiplexadas (MIB)	Recebe um sinal analógico de um equipamento de campo através de um módulo de terminais (FIM) e prepara os dados para que possam ser utilizados pelo resto do sistema. Realiza endereçamento de sinais de entrada analógicos, condicionamento de sinal, sendo possível a escolha de entradas de temperatura, gera alarmes e traça gráficos de tendência.

Tipo	Descrição
Bloco de entrada “smart” (SIB)	Permite receber sinais provenientes de um equipamento de campo inteligente para utilização em outros blocos. Permite modificar a configuração dos parâmetros do transmissor. Este bloco é geralmente utilizado para entradas de equipamentos com comunicação HART.
Bloco de entrada de valores (VIB)	Este bloco recebe sinais digitais de um transmissor inteligente para uso de outros blocos. Permite a medição de mais que uma variável de processo, caso o equipamento assim o permita.
Bloco de saída HART (HOB)	Recebe dados dos blocos de controlo e com base neles prepara um conjunto de sinais analógicos e discretos para enviar a um equipamento inteligente. Partilha as características dos blocos AOB e SIB.
Bloco de autómato programável (PLCB)	Lê ou escreve o conteúdo de um registo de um <i>PLC</i> . Pode também ler ou escrever dados num registo de um <i>PLC</i> .
Bloco de entrada discreta (DIB)	Recebe um sinal digital proveniente de um painel de terminais de entradas discretas e prepara o sinal para que possa ser utilizado pelo resto do sistema. Permite endereçamento e condicionamento de sinais discretos, registo de eventos, gerar alarmes e traçar gráficos de tendência.
Bloco de saída discreta (DOB)	Prepara um estado discreto específico para ser enviado para o campo. Permite armazenamento e endereçamento de sinais discretos, bem como uma saída invertida em relação à entrada.

Tabela 4.2 - Blocos de controlo - funções contínuas - do sistema RS3.

Tipo	Descrição
MAN	Manual (definido pelo utilizador)
P	Controlo proporcional
I	Controlo integral
D	Controlo derivativo
PI	Controlo proporcional e integral
ID	Controlo integral e derivativo
PD	Controlo proporcional e derivativo
PID	Controlo proporcional, integral e derivativo
IB	Integral com BIAS
LL	Lead/Lag
DT	Tempo morto
TOT	Totalizador
TOTSP	Totalizador com referência
RB	Ratio/Bias
SS	Selector de sinal
VLIM	Limitador de velocidade
POLY	Polinómio de sétima ordem
PLI	Piecewise Linear Interpolator
MATH	Definido pelo utilizador
ATPID	Controlo proporcional, integral e derivativo com auto ajuste

Tabela 4.3 - Blocos de controlo - funções discretas - do sistema RS3.

Tipo	Descrição
DISC	Lógica discreta (definido pelo utilizador)
DMC	Controlo de motores
DASMC	Controlo de motores com arranque em sequência
DDSMC	Controlo de motores de duas velocidades
DVC	Controlo de válvulas
DASVC	Controlo de válvulas em sequência
DMVC	Controlo de válvulas motorizadas
RBL	<i>Rosemount Basic Language Control</i>

Anexo 2

Blocos função do sistema *DeltaV*

Tabela 4.4 - Biblioteca de blocos de função do sistema *DeltaV* (versão 9.3.1).

Tipo	Nome	Descrição
Blocos de Entrada e Saída	<i>Alarm Detection</i>	Permite especificar alarmes em parâmetros obtidos de entradas ou provenientes de resultados de outros blocos. Os parâmetros gerados por este bloco podem posteriormente ser utilizados pelo <i>DeltaV Operate</i> para gerar alarmes.
	<i>Analog Input</i>	Acede ao valor medido num canal de entrada/saída.
	<i>Analog Output</i>	Atribui o seu valor de saída a um canal de entrada/saída.
	<i>Discrete Input</i>	Acede a um valor discreto, podendo ser configurado para gerar alarmes e inversão do sinal de entrada.
	<i>Discrete Output</i>	Escreve num canal de entrada/saída um valor de referência predefinido pelo utilizador ou definido pela saída de outro módulo, permitindo ainda fazer inversão do sinal.
	<i>Fieldbus Multiple Discrete Input</i>	Combina as oito entradas de uma carta de entradas discretas e disponibiliza-as a outros blocos sob a forma de uma entrada de oito bits.
	<i>Fieldbus Multiple Discrete Output</i>	Coloca um valor de referência de oito bits nos oito canais de entrada/saída de uma carta de saídas discretas.
	<i>Multiplexed Analog Input</i>	Permite processar até oito medições de um equipamento <i>Fieldbus</i> .
	<i>Pulse Input</i>	É utilizado para fazer contagem de impulsos.

Tipo		Nome	Descrição
Blocos de Controlo Analógico		<i>Bias/Gain</i>	Gera uma saída cujo valor é a entrada acrescida de um offset e por sua vez multiplicada por um ganho.
		<i>Calculation/Logic</i>	Permite ao utilizador especificar uma expressão em texto estruturado para determinar as saídas. Estes blocos podem ter até 16 entradas e 16 saídas.
		<i>Control Selector</i>	Coloca na saída uma de três entradas associadas a controladores PID. A escolha da entrada seleccionada depende do tipo de selecção que pode ser a do valor mais elevado a do valor menos elevado ou a do valor central.
		<i>Deadtime</i>	Introduzir um atraso no sinal de entrada.
		<i>Filter</i>	Faz a filtragem de sinais ruidosos tornando-os mais constantes.
		<i>Input Selector</i>	Coloca na saída uma de quatro entradas mediante um critério de escolha (máximo, mínimo, média).
		<i>Fieldbus Input Selector Extended</i>	Semelhante ao anterior mas em vez de quatro entradas possui oito.
		<i>Lead/Lag</i>	Introduz um avanço, um atraso ou ambos.
		<i>Limit</i>	Limita o valor de entrada entre dois valores pré-definidos.
		<i>Manual Loader</i>	Permite ao utilizador definir o valor de saída do bloco.
		<i>PID</i>	Implementa um controlador PID.
		<i>Ramp</i>	Gera uma rampa crescente ou decrescente. É utilizado p.ex. em arranque suave de motores.
		<i>Rate Limit</i>	Limita a taxa de variação do sinal de saída a um valor pré-definido.
		<i>Ratio</i>	Coloca na saída o valor da entrada primária multiplicada pela razão pretendida entre as duas entradas.
		<i>Scaler</i>	Para efectuar mudanças de escala.
		<i>Signal Characterizer</i>	Permite fazer linearizações de sinais sendo necessário definir até vinte pontos da função que se pretende linearizar.
		<i>Signal Generator</i>	Gera um sinal que simula um processo, utilizando uma combinação específica de uma onda sinusóide com uma onda quadrada, um offset e um valor aleatório.
		<i>Signal Selector</i>	Selecciona o máximo, o mínimo ou a média de até dezasseis entradas.
		<i>Splitter</i>	Através de um sinal de entrada gera dois sinais de saída, com base numa matriz.

Tipo	Nome	Descrição
Blocos Matemáticas	<i>Absolute Value</i>	Devolve o valor absoluto da entrada
	<i>Add</i>	Gera um valor de saída correspondente à soma de até dezasseis entradas.
	<i>Arithmetic</i>	Permite fazer alguns tipos de compensação de fluxos mediante configuração de parâmetros e entradas.
	<i>Comparator</i>	Gera uma saída booleana baseada na comparação de duas entradas. O critério de comparação é configurável.
	<i>Divide</i>	Faz a divisão de uma entrada pela outra.
	<i>Integrator</i>	Integra uma entrada ou então a soma ou subtracção de duas.
	<i>Multiply</i>	Multiplica entre duas a dezasseis entradas.
	<i>Subtract</i>	Subtrai uma entrada à outra.
Blocos Temporizados e Contadores	<i>Counter</i>	A saída fica activa quando a contagem atinge um valor pré-determinado.
	<i>Date Time Event</i>	Fornece funções de data e hora e permite agendar eventos.
	<i>Off-Delay Timer</i>	Atrasa a transição de uma entrada discreta para inactiva por um período de tempo pré-definido.
	<i>On-Delay Timer</i>	Atrasa a transição de uma entrada discreta para activa por um período de tempo pré-definido.
	<i>Retentive Timer</i>	Coloca a saída activa se a entrada estiver activo por um determinado período de tempo.
	<i>Timed Pulse</i>	Gera na saída um impulso de duração pré-determinada quando ocorre uma transição de inactiva para activa da entrada.
Blocos de Contagem de Energia	<i>Flow Metering (AGA_SI)</i>	Concebido para o cálculo de caudais de gás natural medidos por sensores de orifício calibrado ou turbinas. Pode ser utilizado para outro tipo de gases ou líquidos.
	<i>Flow Metering (AGA_US)</i>	Este bloco é igual ao anterior à excepção das unidades utilizadas.
	<i>Isentropic Expansion</i>	Calcula a entalpia final de uma expansão isoentrópica de vapor, baseado na pressão e entropia.
	<i>Saturated Steam Properties - Given Temperature</i>	Calcula a entropia, a entalpia, o volume específico e a pressão de vapor para condições de saturação determinadas pela temperatura.
	<i>Saturated Temperature</i>	Determina a temperatura do vapor, em condições de saturação, com base numa dada pressão.

Tipo	Nome	Descrição
Blocos Lógicos	<i>Steam Density Ratio</i>	Calcula a raiz quadrada da razão entre a densidade do vapor e a densidade correspondente a um fluxo medido com pressão e temperatura calibradas.
	<i>Steam Properties</i>	Determina a entalpia, entropia e volume específico do vapor para a pressão e temperatura correspondente.
	<i>Water Enthalpy</i>	Calcula a entalpia da água em condições de saturação a uma dada temperatura.
	<i>Water Entropy</i>	Determina a entropia da água em condições de saturação a uma dada temperatura.
	<i>Action</i>	Executa uma expressão definida em texto estruturado sempre que a entrada está activa.
	<i>And</i>	Gera uma saída discreta baseada na operação lógica AND de entre duas a dezasseis entradas.
	<i>Bi-directional Edge Trigger</i>	A saída é um impulso de duração correspondente à taxa de refrescamento do módulo sempre que ocorre uma transição na entrada (activa - inactiva ou inactiva - activa).
	<i>Boolean Fan Input</i>	Gera uma saída discreta baseada na soma pesada, estado ou operação lógica OR de uma a dezasseis entradas discretas.
	<i>Boolean Fan Output</i>	Descodifica uma entrada binária em bits individuais e gera uma saída discreta para cada bit.
	<i>Condition</i>	Avalia uma expressão booleana declarada em texto estruturado e caso esta seja verdadeira activa a saída, caso contrário desactiva-a.
	<i>Device Control</i>	Implementa uma máquina de estados controlada pelas entradas com base numa tabela de verdade. Frequentemente utilizado para controlo discreto de motores e válvulas.
	<i>Multiplexer</i>	Coloca na saída uma de até dezasseis entradas. A escolha da entrada a colocar na saída é realizada pelo operador ou pela saída de outro bloco.
	<i>Negative Edge Trigger</i>	Gera um impulso com a duração do tempo de refrescamento do módulo sempre que a entrada passe de activa para inactiva.
	<i>Not</i>	Inverte o sinal discreto de entrada.
	<i>Or</i>	A saída é gerada com base na operação lógica OR de até dezasseis entradas.
	<i>Positive Edge Trigger</i>	Gera um impulso com a duração do tempo de refrescamento do módulo sempre que a entrada passe de inactiva para activa.
	<i>Reset/Set Flip-flop</i>	O valor de saída é um valor discreto baseado na operação lógica NOR entre as entradas <i>SET</i> e <i>RESET</i> .

Tipo	Nome	Descrição
Blocos de Controlo Avançado	<i>Set/Reset Flip-flop</i>	Gera uma saída discreta que tem por base a operação lógica <i>NAND</i> entre as entradas <i>SET</i> e <i>RESET</i> .
	<i>Transfer</i>	Selecciona um de dois sinais analógicos de entrada mediante o estado da entrada <i>SELECTOR</i> (activa - <i>IN_1</i> , inactiva - <i>IN_2</i>).
	<i>Diagnostic</i>	Este bloco possibilita monitorizar equipamentos de campo para posterior análise pela aplicação <i>InSight</i> .
	<i>Fuzzy Logic Control</i>	Implementa controladores baseados em lógica difusa.
	<i>Inspect</i>	Este bloco serve para fazer estatísticas da performance de células processuais ou de todo o sistema.
	<i>Lab Entry</i>	Permite ao utilizador fazer entradas de resultados de análises laboratoriais para utilizar em redes neuronais.
	<i>Model Predictive Control</i>	Utilizados em sistemas de controlo avançado (<i>Advanced Process Control</i> - <i>APC</i>)
	<i>Model Predictive Control Process Simulator</i>	
	<i>Model Predictive Control Professional</i>	
	<i>Neural Network</i>	Permite criar sensores virtuais para estimar a saída de um processo baseada nas suas entradas utilizando redes neuronais.
Itens Especiais	<i>Input Parameter</i>	Parâmetro de entrada que pode ser de vários tipos: definido pelo utilizador ou parâmetros de outros módulos.
	<i>Output Parameter</i>	Parâmetro de saída possível de ser utilizado por outros módulos.
	<i>Internal Read Parameter</i>	Variável interna de leitura para uso exclusivo do módulo.
	<i>Internal Write Parameter</i>	Igual ao anterior mas em vez de uma variável de leitura é uma variável de escrita.
	<i>Custom Block</i>	Permite adicionar ao módulo outros módulos já criados.
	<i>Physical Block</i>	Semelhante ao anterior mas o módulo adicionado reside num equipamento <i>Fieldbus</i> .

Anexo 3

Aplicações do sistema *DeltaV*

Tabela 4.5 - Ferramentas de Engenharia do sistema *DeltaV* (versão 9.3.1).

Aplicação	Descrição
<i>Auto Update Service</i>	O serviço de actualização automática serve para determinar quais os ficheiros que devem ser automaticamente transferidos da estação <i>ProfessionalPLUS</i> para as restantes estações depois de um nó do barramento de controlo ser descarregado.
<i>Configuration Assistant</i>	O assistente de configuração ajuda os utilizadores inexperientes a configurar o sistema. Esta aplicação só pode ser executada na estação onde é realizado o histórico, fornecendo uma panóplia de ferramentas para a gestão de dados de histórico.
<i>Continuous Historian Administration</i>	
<i>Control Studio</i>	O <i>Control Studio</i> é utilizado para criar e modificar os módulo de controlo responsáveis pelo controlo processual. Esta aplicação permite ao utilizador construir os módulos de controlo arrastando itens de uma biblioteca para o diagrama e posteriormente interliga-los para definir um algoritmo. O <i>Control Studio</i> suporta um conjunto de conceitos industriais como os <i>Function Block Diagrams</i> e os <i>Structured Function Charts</i> , sendo possível utilizar elementos de ambas as linguagens dentro do mesmo módulo de controlo. Estas linguagens gráficas são baseadas na norma IEC 61131-3 [4], sendo os blocos função desenvolvidos mediante o standard da <i>Foundation Fieldbus</i> .
<i>Database Administrator</i>	Esta aplicação proporciona a um utilizador com privilégios de administrador um conjunto de ferramentas que lhe permitem realizar tarefas de manutenção da base de dados tais como: criar, apagar e copiar bases de dados bem como realizar cópias de segurança das bases de dados existentes.

<i>DeltaV Explorer</i>	O <i>DeltaV Explorer</i> tem um aspecto semelhante ao Windows Explorer. Esta aplicação permite ao utilizador ver toda a estrutura do sistema e definir os seus componentes, tais como: áreas, nós, alarmes e módulos. Esta aplicação é o centro das aplicações de engenharia possibilitando criar, apagar e mover módulos, configurar todo o hardware do sistema, definir tipos e prioridades dos alarmes, bem como executar outras aplicações do sistema.
<i>DeltaV Operate (Configure)</i>	Esta aplicação pode ser executada em dois modos, o de configuração e o modo run. No modo de configuração permite criar ambientes gráficos de alta resolução para monitorização dos processos. É possível adicionar ao interface com o utilizador imagens, texto, animações, gráficos, sons, bem como os chamados dynamos sendo estes gráficos reutilizáveis na construção de múltiplos interfaces.
<i>DeltaV FlexLock</i>	O <i>DeltaV Flex Lock</i> cria dois ambientes de trabalho duais, um para o Windows e outro para o <i>DeltaV</i> , proporcionando assim um ambiente de operação seguro e aberto ao mesmo tempo. O acesso aos ambientes de trabalho depende dos privilégios do utilizador.
<i>Recipe Studio</i>	O Recipe Studio é utilizado para criar e modificar receitas. Uma receita é um conjunto de informação que identifica os ingredientes, as suas quantidades e o equipamento necessário para produzir um produto.
<i>System Alarm Management</i>	Esta aplicação proporciona ao utilizador uma forma simples e eficiente de ver, habilitar, desabilitar e definir limites e prioridades para múltiplos alarmes.
<i>System Preferences</i>	Esta aplicação permite personalizar as ferramentas de engenharia de modo a possuírem apenas as funções requeridas pelo utilizador sem necessidade de adicionar ou remover software.
<i>User Manager</i>	Com esta ferramenta é possível especificar níveis de acesso de grupos ou indivíduos com diferentes privilégios.

Tabela 4.6 - Aplicações de operação sistema *DeltaV* (versão 9.3.1).

Aplicação	Descrição
<i>Batch History View</i>	Esta aplicação permite interagir com a base de dados de histórico do Batch, possibilitando ver os dados em vários formatos e adicionar comentários ao histórico do Batch.
<i>Batch Operator Interface</i>	O Batch Operator Interface é o interface gráfico utilizado pelo operador para interagir com as operações de Batch, proporcionando diversas vistas do processo.
<i>DeltaV Login</i>	Esta aplicação permite aos operadores entrar e sair no sistema através de registo do nome de utilizador e palavra passe, para além disso permite visualizar qual o operador que está registado e alterar a palavra passe.
<i>Diagnostics</i>	Esta ferramenta fornece informação sobre o estado e integridade dos equipamentos que compõem o sistema.
<i>DeltaV Operate (Run)</i>	O <i>DeltaV Operate</i> em modo run é o interface gráfico entre o operador e o controlo processual,

	permitindo a sua supervisão.
MPC Operate e MPC Operate Pro	Controlo avançado. Implementam modelos preditivos.
Process History View	Esta aplicação apresenta dados de tempo real assim como de histórico sobre a forma de gráficos bem como eventos sobre forma de tabelas. O <i>Process History View</i> é geralmente utilizado para avaliar o comportamento de um processo no tempo.

Tabela 4.7 - Aplicações de controlo avançado do sistema *DeltaV* (versão 9.3.1).

DeltaV Neural	Esta aplicação proporciona um conjunto de ferramentas utilizadas para implementar redes neuronais, permitindo assim criar sensores virtuais para monitorização ou previsão de parâmetros que de outra forma seriam muito difíceis medir.
DeltaV Predict	O <i>DeltaV Predict</i> permite implementar controlo preditivo em pequenos e médios processos.
DeltaV Predict Pro	Esta ferramenta é semelhante à anterior mas permite a sua utilização em processos multi-variável mais complexos.
DeltaV Simulate Suite	Esta aplicação permite instalar o <i>DeltaV</i> num <i>PC</i> vulgar, permitindo fazer todo o tipo de configurações off-line sem necessidade do hardware. Este tipo de estações de trabalho é geralmente utilizado para desenvolvimento <i>off-line</i> e treino de operadores.
Inspect with InSight	A ferramenta <i>Inspect with InSight</i> permite identificar <i>loops</i> com fraco desempenho bem como equipamento de campo com mau funcionamento.
Tune with InSight	O <i>Tune with InSight</i> é uma ferramenta para fazer <i>tuning</i> de <i>loops</i> de controlo, podendo este ser realizado de duas formas, por ordem do utilizador ou adaptativo, sendo este último baseado em algoritmos de auto-aprendizagem.

Anexo 4

Bloco *DMC* do *RS3*

Tabela 5.2 - Configuração de opções do bloco *DMC* do *RS3*.

Opções	Descrição
<i>Confirm Off</i>	Se <i>Yes</i> , <i>@h</i> deverá ser <i>True</i> num período de tempo definido no <i>link</i> contínuo <i>G</i> , após um comando de paragem (passo lógico <i>b</i> activo). Caso não aconteça um alarme soará. Se <i>No</i> , utilizará a negação da confirmação de arranque, através do <i>link @g</i> .
<i>Ignore Confirm Off</i>	Se <i>Yes</i> , não será activa qualquer alarme, mesmo que não seja recebida a confirmação de paragem do equipamento.
<i>Ignore Confirm On</i>	Se <i>Yes</i> , não será activa qualquer alarme, mesmo que não seja recebida a confirmação de arranque do equipamento.
<i>Ignore Interlock</i>	Se <i>Yes</i> o equipamento poderá ser operado mesmo que o <i>link @k</i> seja falso (se verdadeiro o <i>interlock</i> não está activo).
<i>Interlock</i>	Se <i>Yes</i> , <i>@k</i> tem que ser verdadeiro para que o equipamento possa ser operado, independentemente do modo do bloco (Manual ou Automático).
<i>MCC Alarm</i>	Se <i>Yes</i> o passo lógico <i>l</i> será utilizado como indicação de alarme de falta de alimentação ao equipamento. Neste caso é necessário que a opção <i>Confirm Off</i> esteja em <i>Yes</i> .
<i>Retry</i>	Se <i>Yes</i> , o número de tentativas sem sucesso para que seja cumprida a ordem de arranque do equipamento. Caso seja ultrapassado o número de tentativas deverá ser aguardado um período de tempo definido no <i>link</i> contínuo <i>J</i> para nova tentativa.

Security Lockup

Se *Yes*, é necessário um comando de *reset* ao equipamento (@I), independentemente do modo do mesmo, após um estado de falha (arranque, paragem, *interlock* ou *shutdown*).
Se *No*, um simples comando de paragem liberta o equipamento para operação assim que o sinal de falha não esteja activo.

Trip Delay

Se *Yes*, a verificação por parte do bloco de um estado de falha é atrasado por um período de tempo definido no *link* contínuo *F*.
Se *No*, o estado de falha é imediatamente transmitido.

Tabela 5.3 - *Links* discretos e passos lógicos do bloco *DMC* do *RS3*.

<i>Input</i>	Função quando o <i>input</i> é Verdadeiro	<i>Output</i>	Condição para que o <i>output</i> seja verdadeiro
@a	Arranque manual	a	Arranque
@b	Paragem manual	b	Paragem
@c	Configurável	c	Configurável
@d	Arranque automático	d	Seleção para ignorar eventuais opções activas
@e	Paragem automática	e	Condição de falha: activado um ou mais <i>links f, g, h, j, ou k</i> .
@f	Configurável	f	Mais de um sinal de confirmação está activo
@g	Confirmação de arranque	g	Falha no arranque
@h	Confirmação de paragem	h	Falha na paragem
@i	Configurável	i	Configurável
@j	Paragem independentemente do modo do bloco	j	Mudança de estado sem ter sido dado comando (falha)
@k	OK para arranque (quando opção de <i>interlock</i> seleccionada)	k	Equipamento bloqueado. Espera reset
@l	Liberta <i>Security Lockup</i> (quando opção de <i>security lockup</i> seleccionada)	l	Se a opção MCC Alarm estiver seleccionada, a saída deste passo lógico é verdadeira no caso de falha de alimentação ao equipamento. Nas restantes opções é configurável
@m	Configurável	m	Configurável
@n	Configurável	n	Configurável
@o	Configurável	o	Configurável
		p	Configurável

Tabela 5.4 - *Links* contínuos do bloco *DMC* do *RS3*.

Input	Função
<i>A</i>	Configurável
<i>B</i>	Configurável
<i>C</i>	Configurável
<i>D</i>	Configurável
<i>E</i>	Configurável
<i>F</i>	Tempo para arranque. Contagem desde o comando de arranque e a confirmação de equipamento em serviço
<i>G</i>	Tempo para paragem. Contagem desde o comando de paragem e a confirmação do equipamento parado
<i>H</i>	Pré-configurado como Timer mas não é usado na função <i>DMC</i> . Não deve, no entanto, ser utilizado
<i>I</i>	Contagem de tentativas de comando. Usado quando a opção de <i>Retry</i> foi seleccionada
<i>J</i>	Contagem de tempo para equipamento fora de serviço. Usado quando a opção de <i>Retry</i> foi seleccionada
<i>K</i>	Configurável
<i>L</i>	Configurável
<i>M</i>	Configurável
<i>N</i>	Configurável
<i>O</i>	Configurável

Anexo 5

Ficheiro de Especificação de Formato

Ficheiro de Especificação de formato Serial_Dataset.fmt editado com ferramenta de texto e respectiva

```
begin_format_defn Serial_Dataset
```

```
separator \t
```

comment character #

header

```
description,ctrlr,card_slot,card_type,redundant,device_signal_tag,serial_port_slot,serial_port_pio_name,serial_device_index,serial_device_description,serial_device_address,dataset_number,dataset_data_direction,dataset_datatype,dataset_num_values,dataset_output_mode,dataset_output_readback,dataset_PLC_datatype,dataset_PLC_reg_addr_offset
```

```
must exist ctrlr,card slot
```

```
description,ctrlr,card_slot,card_type,redundant,device_signal_tag,serial_port_slot,serial_por  
t_pio_name,serial_device_index,serial_device_description,serial_device_address,dataset_nu  
mber,dataset_data_direction,dataset_datatype,dataset_num_values,dataset_output_mode,d  
ataset_output_readback,dataset_PLC_datatype,dataset_PLC_reg_addr_offset
```

is_a_string_parameter F,F,F,F,F,F,F,F,F,F,F,F,F,F,F,F,F,F

```
end format defn
```

Formatação Inicial

Cabeçalho

Parâmetros

Anexo 6

Parâmetros da Aplicação

Tabela 6.2 - Parâmetros do ficheiro texto e sua proveniência - Criação de *datasets*.

Parâmetro	Descrição	Introdução do Utilizador?	Valor por Defeito
<i>Description</i>	Descrição do <i>dataset</i>	Sim	Nulo
<i>Ctrlr</i>	Nome do controlador de <i>DeltaV</i> onde está fisicamente ligado o <i>PLC</i>	Sim	Nulo
<i>Card_slot</i>	Número da carta de <i>DeltaV</i> onde será ligado o <i>PLC</i> (de 1 a 64)	Sim	Nulo
<i>Card_type</i>	Tipo de carta de <i>DeltaV</i>	Não	<i>SERIAL_32DS_SER2</i>
<i>Redundant</i>	Carta redundante ou não	Não	F
<i>Device_signal_tag</i>	<i>Tag</i> pelo qual irão ser identificados os registos	Sim ⁽¹⁾	⁽²⁾
<i>Serial_port_slot</i>	Porta da carta <i>DeltaV</i> onde será ligado o <i>PLC</i> (porta 1 ou 2)	Sim	1
<i>Serial_port_pio_name</i>	Denominação da porta	Não	<i>SERIAL_PORT</i>
<i>Seria_device_index</i>	Número do <i>PLC</i> no caso de existir mais do que um	Não	1
<i>Serial_device_description</i>	Descrição do <i>PLC</i>	Sim	Nulo
<i>Serial_device_address</i>	Número do <i>PLC</i> proveniente do <i>CS for RS3</i>	Não	⁽³⁾
<i>Dataset_number</i>	Número de preenchimento sequencial correspondente ao número do <i>dataaset</i>	Não	⁽⁴⁾
<i>Dataset_data_direction</i>	Indicação de leitura ou escrita	Não	⁽³⁾

<i>Dataset_datatype</i>	Indicação do tipo de dado a receber ou enviar	Não	(5)
<i>Dataset_num_values</i>	Contagem do número de registos	Não	(3)
<i>Dataset_output_mode</i>	Quando em modo de escrita permite escrever um valor individual ou um conjunto de valores	Não	<i>Complete block</i>
<i>Dataset_output_reedback</i>	Redireccionamento da escrita	Não	F
<i>Dataset_PLC_datatype</i>	Tipo de registo pertencente a uma determinada gama (tabela 6.1)	Não	(3)
<i>Dataset_PLC_reg_addr_offset</i>	Valor do offset relativa à gama do registo em questão	Não	(4)

(1) - O utilizador introduz o primeiro conjunto de letras. O programa acrescenta *_READ* no caso de registos de leitura ou *_WRITE* no caso dos de escrita. Após a indicação da direcção no conjunto de registos é adicionado *_FLAGS* para valores discretos e *_UINT* para valores analógicos. É ainda adicionado um carácter numérico para distinguir conjuntos de registos com as mesmas duas características referidas anteriormente.

(2) - O programa não prossegue se este campo estiver vazio.

(3) - Valor proveniente do *CS for RS3*.

(4) - O programa preenche automaticamente.

(5) - O programa preenche o valor segundo a tabela 6.3 de conversão de dados de *CS for RS3* e o correspondente no *DeltaV*.

Tabela 6.3 - Endereços *MODBUS* e respectiva descrição no *DeltaV*.

Direcção	<i>RS3 Data Type</i>	<i>String</i> Correspondente
Leitura	<i>Flags</i>	<i>BOOLEAN_WITH_STATUS</i>
Leitura	<i>U Int</i>	<i>UINT16_WITH_STATUS</i>
Escrita	<i>Flags</i>	
Escrita	<i>U Int</i>	

Anexo 7

Parâmetros da Aplicação

Tabela 6.3 - Parâmetros dos ficheiros *.FHX* e sua proveniência.

	Parâmetro	Descrição	Introdução do Utilizador?	Valor por Defeito
Parâmetro comuns a todos os módulos	<i>Attribute_Instance Name</i>	Referente aos parâmetros utilizados	Não	Param1
	<i>Controller</i>	Nome do controlador de <i>DeltaV</i>	Sim	Nulo
	<i>Function_Block Name</i>			
	<i>Primary_Control_Display</i>	Nome do sinóptico de operação	Sim	Nulo
	<i>Text</i>	Texto dentro do módulo	Não	(1)
	<i>Value{Ref}</i>	Referência a um dataset ou a outro módulo	Não	(1)
Landing module tipo Write	<i>Attribute Name</i>	Nome módulo	Sim	
	<i>Category</i>	Categoria <i>DeltaV</i>	Não	(1)
	<i>Connection</i>	Existência de ligação a outros blocos	Não	(1)
	<i>Module Tag</i>	Tag	Sim	
	<i>Rectangle</i>	Rectângulo que contém o parâmetro	Não	(1)
	<i>Type</i>	Tipo de módulo no <i>DeltaV</i>	Não	(1)
	<i>Value{Ref}</i>	Referência externa	Não	(1)

	Parâmetro	Descrição	Introdução do Utilizador?	Valor por Defeito
Landing tipo Write_1	<i>Description</i>	Descritivo	Sim	
	<i>Destination</i>	Destino do módulo no <i>DeltaV</i>	Não	(1)
	<i>Extensible Attribute</i>	Atributos em quantidade variável	Não	(1)
	<i>Function Block Name: BFI</i>	Bloco de função <i>BFI</i>	Não	(1)
	<i>ID</i>	Identificação do módulo no <i>DeltaV</i>	Não	(1)
	<i>Module Tag</i>	Tag	Sim	
	<i>Rectangle</i>	Rectângulo que contém o parâmetro	Não	(1)
	<i>Value{Ref}</i>	Referência externa	Não	(1)
	<i>Wire Source</i>	Conector entre parâmetros e/ou blocos de função	Não	(1)
Landing module tipo Read	<i>Category</i>	Categoria <i>DeltaV</i>	Não	(1)
	<i>Connection</i>	Existência de ligação a outros blocos	Não	(1)
	<i>Module Tag</i>	Tag	Não	(1)
	<i>Rectangle</i>	Rectângulo que contém o parâmetro	Não	(1)
	<i>Type</i>	Tipo de módulo no <i>DeltaV</i>	Não	(1)
	<i>Value{Ref}</i>	Referência externa	Não	(1)
Landing module tipo Read_1	<i>Description</i>	Descritivo	Não	(1)
	<i>Destination</i>	Destino do módulo no <i>DeltaV</i>	Não	(1)
	<i>Extensible_Attribute</i>	Atributos em quantidade variável	Não	(1)
	<i>Function_Block Name: BFO</i>	Bloco de função <i>BFO</i>	Não	(1)
	<i>ID</i>	Identificação do módulo no <i>DeltaV</i>	Não	(1)
	<i>MODULE TAG</i>	Tag	Não	(1)
	<i>RECTANGLE</i>	Rectângulo que contém o parâmetro	Não	(1)
	<i>WIRE SOURCE</i>	Conector entre parâmetros e/ou blocos de função	Não	(1)
	<i>Value{Ref}</i>	Referência externa	Não	(1)

(1) - O programa preenche automaticamente.

Anexo 8

Fase I - Testes Efectuados

Tabela 7.1 - Fase I: Testes efectuadas para a criação de *datasets*.

		Teste 1	Teste 2	Teste 3
		FOB PTG4-070 LIT25518A LIT2553D Sim	FOB PTG4-070 SCHNEIDER1 SCHNEIDER4 Sim	FOB PTG4-070 FGFOB-01 FGFOB-13 Não
Registos			40001 - 40003 - 3 registos 40004 - 1 registo	5001 - 5208 (208 registos) 1 - 16 registos
Ficheiro TXT	linha 1	READ_FLAGS WRITE_FLAGS READ_UINT dataset_data_direction dataset_datatype dataset_num_values dataset_PLC_datatype dataset_PLC_reg_addr_offset	40001 - 40080 (80 registos) INPUT UINT16_WITH_STATUS 80 HOLDING REGISTERS 0	OUTPUT UINT16_WITH_STATUS 16 COILS 0
	linha 2	dataset_data_direction dataset_datatype dataset_num_values dataset_PLC_datatype dataset_PLC_reg_addr_offset	OUTPUT UINT16_WITH_STATUS 1 HOLDING REGISTERS 3	INPUT BOOLEAN_WITH_STATUS 100 COILS 5000
	linha 3	dataset_data_direction dataset_datatype dataset_num_values dataset_PLC_datatype dataset_PLC_reg_addr_offset		INPUT BOOLEAN_WITH_STATUS 100 COILS 5100
	linha 4	dataset_data_direction dataset_datatype dataset_num_values dataset_PLC_datatype dataset_PLC_reg_addr_offset		INPUT BOOLEAN_WITH_STATUS 8 COILS 5200
	linha 5	dataset_data_direction dataset_datatype dataset_num_values dataset_PLC_datatype dataset_PLC_reg_addr_offset		

Tabela 7.2 - Fase I: Testes efectuadas para a criação de *datasets* (continuação).

		Base de Dados Projecto Tag inicial Tag final Consecutivo?	Teste 4 FUT PTG2-039 F4401-1 FIQ4401-SP Sim
Registos		READ_FLAGS READ_FLAGS READ_FLAGS WRITE_FLAGS READ_UINT READ_UINT WRITE_UINT WRITE_UINT	41001-41005 - 5 registos 41011-41013 - 3 registos 41101-41102 - 2 registos 41501-41502 - 2 registos 42001-42010 - 10 registos 42750 - 1 registo 42701-42712 - 12 registos 42751 - 1 registo
Ficheiro TXT	linha 1	dataset_data_direction dataset_datatype dataset_num_values dataset_PLC_datatype dataset_PLC_reg_addr_offset	INPUT BOOLEAN_WITH_STATUS 100 HOLDING REGISTERS 1000
	linha 2	dataset_data_direction dataset_datatype dataset_num_values dataset_PLC_datatype dataset_PLC_reg_addr_offset	INPUT BOOLEAN_WITH_STATUS 2 HOLDING REGISTERS 1100
	linha 3	dataset_data_direction dataset_datatype dataset_num_values dataset_PLC_datatype dataset_PLC_reg_addr_offset	OUTPUT UINT16_WITH_STATUS 2 HOLDING REGISTERS 1500
	linha 4	dataset_data_direction dataset_datatype dataset_num_values dataset_PLC_datatype dataset_PLC_reg_addr_offset	INPUT UINT16_WITH_STATUS 10 HOLDING REGISTERS 2000
	linha 5	dataset_data_direction dataset_datatype dataset_num_values dataset_PLC_datatype dataset_PLC_reg_addr_offset	INPUT UINT16_WITH_STATUS 1 HOLDING REGISTERS 2749
	linha 6	dataset_data_direction dataset_datatype dataset_num_values dataset_PLC_datatype dataset_PLC_reg_addr_offset	OUTPUT UINT16_WITH_STATUS 12 HOLDING REGISTERS 2700
	linha 7	dataset_data_direction dataset_datatype dataset_num_values dataset_PLC_datatype dataset_PLC_reg_addr_offset	OUTPUT UINT16_WITH_STATUS 1 HOLDING REGISTERS 2750

Anexo 9

Fase II - Testes Efectuados

Tabela 7.3 - Fase I: Testes efectuadas para a criação de ficheiros *FHX*.

		Teste 1	Teste 2	Teste 3	Teste 4
	Base de Dados Projecto Tag inicial Tag final Consecutivo?	FOB PTG4-070 LIT25518A LIT2553D Sim	FOB PTG4-070 SCHNEIDER1 SCHNEIDER4 Sim	FOB PTG4-070 FGFOB-01 FGFOB-13 Não	FUT PTG2-039 F4401-1 FIQ4401-SP Sim
Registos	<i>READ_FLAGS</i> <i>READ_FLAGS</i> <i>READ_FLAGS</i> <i>WRITE_FLAGS</i> <i>READ_UINT</i> <i>READ_UINT</i> <i>WRITE_UINT</i> <i>WRITE_UINT</i>	40001 - 40080 (80 registos)	40001 - 40003 - 3 registos 40004 - 1 registo	5001 - 5208 (208 registos) 1 - 16 registos	41001-41005 - 5 registos 41011-41013 - 3 registos 41101-41102 - 2 registos 41501-41502 - 2 registos 42001-42010 - 10 registos 42750 - 1 registo 42701-42712 - 12 registos 42751 - 1 registo
Ficheiro <i>FHX</i>	tipo <i>READ</i> tipo <i>READ_1</i> tipo <i>WRITE</i> tipo <i>WRITE_1</i> tipo <i>DI</i> tipo <i>AI</i>	2 (50; 30 parâmetros) 80	1 (3 parâmetros) 1 (1 parâmetro) 31	5 (50; 50; 50; 50; 8 parâmetros) 1 (16 parâmetros) 123	2 (10; 12 parâmetros) 3 (5; 3; 2 parâmetros) 1 (2 parâmetros) 1 (2 parâmetros) 130 10