

RICARDO JORGE SILVÉRIO DE MAGALHÃES MACHADO

Metodologias de Desenvolvimento em Projectos de Engenharia de Computadores no Suporte à Implementação de Sistemas de Informação Distribuídos Não Convencionais (Industriais)

Tese de Doutoramento submetida à ESCOLA DE ENGENHARIA DA UNIVERSIDADE DO MINHO
para a obtenção do grau académico de
Doutor em *Informática/Engenharia de Computadores*.



UNIVERSIDADE DO MINHO
ESCOLA DE ENGENHARIA
DEPARTAMENTO DE SISTEMAS DE INFORMAÇÃO

Braga, Novembro de 2000

ESCOLA DE ENGENHARIA DA UNIVERSIDADE DO MINHO
DEPARTAMENTO DE SISTEMAS DE INFORMAÇÃO

*GRUPO DE ENGENHARIA DE TELECOMPUTAÇÃO
CENTRO DE INVESTIGAÇÃO ALGORITMI*

**METODOLOGIAS DE DESENVOLVIMENTO EM
PROJECTOS DE ENGENHARIA DE COMPUTADORES NO
SUPORTE À IMPLEMENTAÇÃO DE
SISTEMAS DE INFORMAÇÃO DISTRIBUÍDOS
NÃO CONVENCIONAIS (INDUSTRIAIS)**

Ricardo Jorge Silvério de Magalhães Machado

Licenciado em *Engenharia Electrotécnica e de Computadores*
Ramo de *Telecomunicações e Computadores*
pela FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO, 1994

Mestre em *Informática*
Área de Especialização em *Sistemas Distribuídos,*
Comunicações por Computador e Arquitectura de Computadores
pela ESCOLA DE ENGENHARIA DA UNIVERSIDADE DO MINHO, 1996

Tese de Doutoramento submetida para satisfação dos requisitos do
Grau Académico de Doutor em *Informática/Engenharia de Computadores*
e realizada sob a supervisão científica do Prof. Doutor Eng.º Henrique Manuel Dinis dos Santos,
Professor Associado do DEPARTAMENTO DE SISTEMAS DE INFORMAÇÃO DA
ESCOLA DE ENGENHARIA DA UNIVERSIDADE DO MINHO

Braga, Novembro de 2000

FICHA TÉCNICA

Título: Metodologias de Desenvolvimento em Projectos de Engenharia de Computadores no Suporte à Implementação de Sistemas de Informação Distribuídos Não Convencionais (Industriais)

Autor: Ricardo Jorge Silvério de Magalhães Machado

Âmbito: Tese de Doutoramento (ramo de Informática, área de conhecimento em Engenharia de Computadores)

Orientador: Henrique Manuel Dinis dos Santos

Instituição: Escola de Engenharia - Universidade do Minho

Data de Início: Projecto de Doutoramento aprovado pelo Conselho Científico da Escola de Engenharia da Universidade do Minho em 1997/07/23

Data de Conclusão: Novembro de 2000

Impressão e Acabamentos: Serviços de Reprografia e Publicações, Universidade do Minho, Largo do Paço, 4700-320 Braga

Edição: 1ª edição (8 exemplares), Braga, Novembro de 2000

2ª edição (92 exemplares), Braga, Abril de 2001

Copyright: Esta publicação pode ser reproduzida ou transmitida por qualquer forma ou por qualquer processo electrónico, mecânico ou fotográfico (incluindo fotocópia, xerocópia ou gravação) sem autorização prévia e escrita do autor. A eventual transcrição de pequenos textos ou passagens é autorizada, desde que devidamente referenciada. No entanto, esta autorização não deve, de modo nenhum, ser interpretada como sendo extensiva à transcrição de textos em recolhas antológicas, ou similares, donde resulte prejuízo para o interesse desta publicação.

*À Sofia e à pequena Inês,
À mãe Lourdes e ao pai Manuel.*

*“A verdade de um curso não está no que aí se aprende, mas no que disso sobeja:
o halo que isso transcende e onde podemos achar-nos homens.”*

Virgílio Ferreira

Entidades, Programas e Iniciativas que Apoiaram esta Tese:



Departamento de Informática



PRAXIS

Resumo

Os sistemas (baseados em computador) tempo-real embebidos constituem, cada vez mais, um dos suportes computacionais privilegiados para ajudar a dotar a almejada *sociedade da informação e do conhecimento* das necessárias infra-estruturas tecnológicas, caracterizadas pela versatilidade funcional (reconfiguração dinâmica), ubiquidade (distribuição generalizada e computação arbitrariamente remota) e heterogeneidade implementacional (*hardware/software*). Perante a recente proliferação e apetência por este tipo de sistemas, cada vez mais complexos, computacionalmente poderosos e não convencionais, os projectistas têm sido confrontados com a “inexistência” (fraca existência) de metodologias de desenvolvimento compatíveis com as exigências de projecto que aquele tipo de sistemas coloca, nomeadamente no que diz respeito ao controlo da complexidade e à garantia da continuidade dos modelos.

Esta tese propõe uma metodologia de desenvolvimento de sistemas tempo-real embebidos, designada de *INES (integrated environment for embedded systems design)*, que recorre a especificações homogéneas (representações unificadas), neutras (independência da implementação), orientadas por objecto, multi-vista e parcialmente compatíveis com a norma *OMG UML*, sobre as quais actua o processo de reificação que adopta a proposta da estratégia do *4-set rule set* para garantir a continuidade dos modelos na passagem dos requisitos do utilizador para os requisitos do sistema. Para suportar especificamente as vistas dinâmicas é proposto um meta-modelo baseado em redes de Petri orientadas por objectos, o meta-modelo *shobi-PN v2.0*.

No que diz respeito ao macroprocesso da metodologia *INES*, é sugerida uma abordagem em que existem três níveis distintos de co-projecto *hardware/software* (3 microprocessos independentes), como forma de promover a minimização do tempo de implementação dos sistemas (soluções finais), por reutilização intensiva e parametrização de módulos, designados de FMOTSS (*functional-modules-off-the-shelf*), previamente desenvolvidos e afectos a classes de funcionamento perfeitamente caracterizadas. No âmbito dos microprocessos de níveis 2 e 3 da metodologia *INES*, a formalização das interfaces dos FMOTSS pode ser realizada à custa da utilização da notação gráfica FIDL (*FMOTSS interface definition language*) proposta e a concepção arquitectural é baseada na utilização do padrão (*design pattern*) *Multi-Level ICIS* proposto.

Adicionalmente à metodologia *INES*, foram propostos (adoptados e adaptados) dois ambientes de desenvolvimento: (1) um que suporta o nível 2 de (co-)projecto (desenvolvimento de FMOTSS) à custa da ferramenta de CASE (*computer aided software engineering*) *OBLOG*, para a qual foi desenvolvido um pacote de geração automática de código *C* para a família *MCS-51* da *INTEL* e adaptado um pacote de geração automática de código *JAVA* para proceder à simulação funcional de FMOTSS; (2) outro que suporta o nível 3 de (co-)projecto (desenvolvimento de soluções finais) à custa da ferramenta de CAE (*computer aided*

engineering) LabVIEW, para a qual foi desenvolvida a biblioteca VAL (*virtual automation library*) para elevar o nível de abstracção do desenvolvimento das soluções finais. Foram, também, desenvolvidos o escalonador VASch (*virtual automation scheduler*), para dotar os FMOTSs da capacidade de multiprocessamento, e o protocolo de comunicações VAP (*virtual automation protocol*), para implementar, sobre uma rede industrial baseada em CAN, um pacote de serviços de comunicações para interligar funcionalmente vários FMOTSs.

Uma vez que os sistemas de informação industriais orientados ao controlo (*industrial control-based information systems - ICISs*) constituem a área aplicacional privilegiada desta tese, foram utilizados dois casos de estudo industriais e reais para validar as propostas metodológicas e tecnológicas aqui sugeridas. No âmbito do primeiro, realizado na *BLAUPUNKT AUTO-RÁDIOS PORTUGAL*, procedeu-se à especificação completa, e sua simulação funcional e estatística, de um controlador distribuído para uma linha de produção de auto-rádios. No âmbito do segundo, realizado na *TÊXTIL A. FALCÃO*, realizou-se a concepção e implementação de um sistema de supervisão de uma linha de produção de meias, que possibilitou o desenvolvimento de um FMOTS capaz de supervisionar diversas marcas/modelos de teares para a indústria têxtil.

Palavras Chave: *metodologias de desenvolvimento, projecto, co-projecto, especificação, redes de Petri, orientação aos objectos, desenvolvimento baseado em componentes, arquitectura de software, padrões de concepção, arquitecturas reconfiguráveis, sistemas tempo-real, sistemas embebidos, sistemas baseados em computador, UML, OBLOG, LabVIEW.*

Áreas Temáticas: *engenharia de software, arquitectura de sistemas informáticos, informática industrial distribuída, sistemas de informação industriais.*

Abstract

Real-time embedded (computer-based) systems have been increasingly used as one of the privileged computational supports to endow the desired *knowledge and information society* with the necessary technological infrastructures, characterised by its functional versatility (dynamic reconfiguration), ubiquity (generalised distribution and arbitrary remote computation) and heterogeneous implementations (hardware/software). In the presence of the recent proliferation and demand for this kind of systems, each time more complex, computationally powerful and non-conventional, designers have been confronted with the “non-existence” (weak existence) of development methodologies capable of dealing with the demanding design requirements that those kind of systems poses, namely in what concerns the control of complexity and of models’ continuity.

This thesis proposes the *INES (integrated environment for embedded systems design)* methodology for real-time embedded systems development, that employs homogeneous (unified representations), neutral (implementation independence), object-oriented, multiple view and *OMG UML* partially compliant specifications. The reification process, executed over those specifications, adopts the proposed *4-step rule set* strategy to assure the models’ continuity in the transformation of user’s requirements into system’s requirements. To support the dynamic views, the author proposes one object-oriented Petri net-based meta-model, called *shobi-PN v2.0*.

In what concerns *INES* macroprocess, a 3-level hardware/software co-design approach (3 independent microprocesses) is suggested, as a means to reach a minimal implementation time of the desired systems (final solutions). This goal is achieved by reusing and parameterising pre-synthesised modules, called FMOTSSs (*functional-modules-off-the-shelf*), belonging to clearly defined functional classes. Within *INES* microprocesses of levels 2 and 3, it is possible to formalise the FMOTSSs interface by using the proposed FIDL (*FMOTSSs interface definition language*) graphical notation, and architectural design is based on the proposed *Multi-Level ICIS* design pattern.

Besides the *INES* development methodology, this thesis has proposed (adopted and adapted) two design environments: (1) one that supports level 2 of (co-)design (development of FMOTSSs) by using the *OBLOG CASE (computer aided software engineering)* tool, for which a package for automatic *C* code generation for the *INTEL MCS-51* family has been developed and a package for automatic *JAVA* code generation has been adapted to allow the functional simulation of FMOTSSs; (2) another that supports level 3 of (co-)design (development of final solutions) by using the *LabVIEW CAE (computer aided engineering)* tool, for the *VAL (virtual automation library)* library which has been developed, to allow the rising of the abstraction level in the development of final solutions. Additionally, the *VASch (virtual automation*

scheduler) scheduler has been developed, to endow FMOTSs with the capability of multiprocessing, and the *VAP (virtual automation protocol)* communications protocol has been defined, to implement a communication service pack, over an industrial network based on *CAN*, to allow the functional interconnect of several FMOTSs.

Since industrial control-based information systems (ICISs) constitutes the main application area of this thesis, two real industrial case studies have been used to validate the methodological and technological proposals suggested here. In the first case, performed at *BLAUPUNKT AUTO-RÁDIOS PORTUGAL*, a distributed controller for a production line of car radios has been fully specified and functionally and statistically simulated. In the second case, performed at *TÊXTIL A. FALCÃO*, a supervision system for a production line of socks has been designed and implemented, which allowed the development of one FMOTS capable of supervising several kinds of knitting machines for the textile industry.

Keywords: *design methodologies, design, co-design, specification, Petri nets, object orientation, component-based design, software architecture, design patterns, reconfigurable architectures, real-time systems, embedded systems, computer-based systems, UML, OBLOG, LabVIEW.*

Thematic Areas: *software engineering, architecture of computer systems, distributed industrial informatics, industrial information systems.*

Índice

Contracapa	i
Ficha Técnica	ii
Dedicatória	iii
Entidades, Programas e Iniciativas que Apoiaram a Tese	iv
Resumo	v
Abstract	vii
Índice	ix
Índice de Figuras	xv
Índice de Tabelas	xxiii
Acrónimos e Abreviaturas	xxv
Prefácio	xxxii
Composição do Júri	xxxv
<i>Parte I: Introdução</i>	1
<hr/>	
1. Introdução ao Projecto em Engenharia de Computadores	3
1.1. Introdução	5
1.1.1. <i>Sistemas, Abordagem Sistémica e Engenharia</i>	7
1.1.2. <i>Sistemas Baseados em Computador</i>	10
1.1.3. <i>Metodologias de Desenvolvimento em Eng^a. de Computadores</i>	11
1.2. Contribuição do Trabalho Desenvolvido	12
1.2.1. <i>Objectivos da Tese</i>	13
1.2.2. <i>Metodologia de Investigação</i>	16
1.2.3. <i>Projectos de I&D</i>	18
1.3. Conteúdo e Organização da Tese	19
1.3.1. <i>Enquadramento e Temáticas de Suporte</i>	21
1.3.2. <i>Propostas Metodológicas e Experimentação</i>	22
<i>Leituras Recomendadas</i>	24
<i>Bibliografia</i>	24

2. Sistemas Tempo-Real	29
2.1. Introdução	31
2.1.1. <i>Sistemas Reactivos</i>	32
2.1.2. <i>Sistemas Tempo-Real</i>	32
2.1.3. <i>Sistemas Embebidos</i>	32
2.2. Evolução dos Sistemas Tempo-Real	33
2.2.1. <i>Hardware</i>	34
2.2.2. <i>Software</i>	34
2.2.3. <i>Metodologias</i>	35
2.3. Sistemas Tempo-Real no Controlo Industrial	36
2.3.1. <i>Tolerância a Faltas</i>	37
2.3.2. <i>Fiabilidade do Software</i>	37
2.3.3. <i>Controlo de Processos e Manufactura</i>	38
2.4. Modelação e Especificação	39
2.4.1. <i>Características dos Sistemas Tempo-Real</i>	40
2.4.2. <i>Metodologia de Especificação</i>	41
2.4.3. <i>Meta-Modelos de Especificação</i>	44
2.4.4. <i>Requisitos Não Funcionais</i>	52
2.5. Metodologias de Desenvolvimento	53
2.5.1. <i>Ciclo de Vida do Sistema</i>	53
2.5.2. <i>Métodos Funcionais</i>	56
2.5.3. <i>Métodos Orientados por Objectos</i>	58
2.6. Conclusões	64
 <i>Leituras Recomendadas</i>	 65
<i>Bibliografia</i>	65
 3. Co-Projecto Hardware/Software	 71
3.1. Introdução	73
3.1.1. <i>Abordagem Orientada à Concepção</i>	74
3.1.2. <i>Abordagem Orientada à Síntese</i>	75
3.1.3. <i>Abordagem Orientada ao Co-Projecto</i>	76
3.1.4. <i>Aplicações do Co-Projecto</i>	77
3.2. Co-Especificação Hardware/Software	79
3.2.1. <i>Comparação Hardware/Software</i>	80
3.2.2. <i>Modelação de Hardware Orientada por Objectos</i>	82
3.3. Co-Síntese Hardware/Software	86
3.3.1. <i>Partição Hardware/Software</i>	87
3.3.2. <i>Escalonamento do Software</i>	89
3.3.3. <i>Síntese Hardware/Software</i>	91
3.3.4. <i>Co-Validação</i>	92

3.4. Metodologias de Desenvolvimento no Co-Projecto	93
3.4.1. Abordagem Orientada à Concepção	95
3.4.2. Abordagem Orientada ao Co-Projecto	97
3.4.3. Ambientes e Metodologias de Co-Projecto	99
3.5 Conclusões	107
<i>Leituras Recomendadas</i>	107
<i>Bibliografia</i>	108
4. Architecturas Reconfiguráveis	113
4.1. Introdução	115
4.1.1. Tecnologias de Implementação de Sistemas Digitais	116
4.1.2. Estratégias de Implementação de Sistemas Digitais	118
4.2. Componentes Lógicos Reconfiguráveis	121
4.2.1. SPLDs	122
4.2.2. CPLDs	124
4.2.3. FPGAs	127
4.2.4. FPIs	133
4.2.5. Tecnologia ISP	133
4.2.6. Taxionomia Revisitada	135
4.3. Sistemas Reconfiguráveis	136
4.3.1. Características Arquitecturais	137
4.3.2. Modos de Funcionamento	143
4.4. Paradigma da Computação Reconfigurável	147
4.4.1. Custom Computing Machines	147
4.4.2. Metodologias e Ferramentas	148
4.5. Conclusões	149
<i>Leituras Recomendadas</i>	150
<i>Bibliografia</i>	150
<i>Parte III: Metodologias e Casos de Estudo</i>	155
<hr/>	
5. Análise, Requisitos e Especificação	157
5.1. Introdução	159
5.1.1. Engenharia de Requisitos	160
5.1.2. Modelo do Processo de Desenvolvimento	162
5.2. Levantamento dos Requisitos	164
5.2.1. Diagramas de Contexto	165
5.2.2. Diagramas de Casos de Uso	168
5.2.3. Diagramas de Objectos	173
5.2.4. Sistema Controlado	178
5.2.5. Diagramas de Sequência	183

5.3. Formalização dos Requisitos	187
5.3.1. Controladores Paralelos	187
5.3.2. Redes de Petri e Sistemas Embebidos	189
5.3.3. Meta-Modelo shobi-PN	191
5.3.4. Meta-Modelo shobi-PN v2.0	194
5.4. Especificação e Validação Funcional	207
5.4.1. Diagramas de Classes	208
5.4.2. Semântica Operacional	211
5.4.3. Estrutura do Repositório	232
5.4.4. Animação e Simulação	236
5.4.5. Documento de Requisitos	238
5.5. Conclusões	238
<i>Leituras Recomendadas</i>	239
<i>Bibliografia</i>	240
6. Concepção e Arquitectura de Sistemas	243
6.1. Introdução	245
6.1.1. Projecto ao Nível do Sistema	246
6.1.2. Níveis de Projecto	249
6.2. Modelo do Processo de Desenvolvimento Revisitado	250
6.2.1. Configurações Organizacionais	250
6.2.2. Diagramas de Casos de Uso ao Nível do Processo	254
6.2.3. Diagramas de Objectos ao Nível do Processo	261
6.2.4. Macroprocesso Revisitado	264
6.3. Desenvolvimento Baseado em Componentes	270
6.3.1. Componentes e Objectos	273
6.3.2. Modelação da Arquitectura Alvo	275
6.3.3. Mecanismos de Reutilização e Interligação	283
6.4. Concepção Arquitectural	291
6.4.1. Padrões de Concepção	291
6.4.2. Generalização Funcional de Componentes	299
6.4.3. Microprocessos de Níveis 2 e 3	314
6.5. Conclusões	320
<i>Leituras Recomendadas</i>	321
<i>Bibliografia</i>	321
7. Síntese e Implementação de Sistemas	325
7.1. Introdução	327
7.1.1. Sistemas Operativos Tempo-Real	328
7.1.2. Linguagens de Alto-Nível	330
7.2. Reconfigurabilidade nas Arquitecturas Alvo	333
7.2.1. Redesenho Arquitectural	334
7.2.2. Integração Metodológica	339

7.3. Síntese e Implementação de FMOTSS	340
7.3.1. Geração Automática de Código	340
7.3.2. Animação em JAVA	346
7.4. Síntese e Implementação de Soluções Finais	348
7.4.1. Ambiente de Desenvolvimento de Nível 3	349
7.4.2. Acesso e Distribuição da Informação em Soluções Finais	355
7.5. Conclusões	360
 <i>Leituras Recomendadas</i>	 361
<i>Bibliografia</i>	361

Parte IV: Conclusões **365**

8. Conclusões e Reflexões Finais **367**

8.1. Análise Crítica	369
8.1.1. Metodologia	370
8.1.2. Ferramentas	371
8.1.3. Casos de Estudo	372
8.2. Trabalho Futuro	373
8.2.1. Tópicos de Investigação	373
8.2.2. Áreas Aplicacionais	374
 <i>Bibliografia</i>	 375

A nexos **377**

A - Documento de Requisitos do Projecto das Linhas HIDRO **379**

B - Gramática da Linguagem OBLOG **497**

C - Classificação dos Equipamentos da Indústria Têxtil **507**

D - Curriculum Vitae **513**

Índice de Figuras

Figura 1.1: Principais fases do ciclo de vida de um projecto.	6
Figura 1.2: A complexidade é proporcional às interacções entre as partes de um sistema.	8
Figura 1.3: Aumento do nível de abstracção.	9
Figura 1.4: Modelação de sistemas segundo múltiplas vistas.	11
Figura 1.5: Níveis CIM numa organização industrial.	14
Figura 1.6: Pirâmide do sistema de informação global.	15
Figura 2.1: Sistemas reactivos tempo-real embebidos.	33
Figura 2.2: Sistema de controlo de processos industriais.	38
Figura 2.3: Sistema de controlo de fabrico.	39
Figura 2.4: Especificação de sistemas.	42
Figura 2.5: Características da metodologia de especificação.	43
Figura 2.6: Especificação FSM de um detector de paridades em STD.	45
Figura 2.7: Diagrama de blocos de uma FSMD genérica.	45
Figura 2.8: Especificação HCFSM de um controlador na linguagem <i>STATECHARTS</i>	45
Figura 2.9: Especificação em PN de um controlador paralelo.	46
Figura 2.10: Especificação de uma expressão aritmética em DFG.	46
Figura 2.11: Especificação em CFG de um detector de erros.	47
Figura 2.12: Especificação de uma arquitectura UMA em CCD.	48
Figura 2.13: Especificação em CDFG de um <i>switch</i> de operações aritméticas.	49
Figura 2.14: Especificação PSM do <i>receive</i> numa UART na linguagem <i>SPECCHARTS</i>	50
Figura 2.15: Classificação de meta-modelos.	51
Figura 2.16: Ciclo de vida do sistema.	54
Figura 2.17: O modelo em cascata.	55
Figura 2.18: O modelo em V.	55
Figura 2.19: O modelo em espiral.	56

Figura 2.20: Especificação em <i>SDRTS</i> de um controlor de PH.	57
Figura 2.21: Estrutura de objectos de um controlador industrial em <i>OOD</i>	60
Figura 2.22: Diagrama de sequência em <i>RTUML</i> de um cenário de controlo de um elevador.	62
Figura 3.1: Abordagem orientada à concepção na implementação de um controlador de rede.	75
Figura 3.2: Abordagem orientada à síntese na implementação de um controlador de rede.	75
Figura 3.3: Compromisso <i>hardware/software</i>	76
Figura 3.4: Abordagem orientada ao co-projecto na implementação de um controlador de rede.	77
Figura 3.5: Esquema de um <i>system-level ASIC</i>	78
Figura 3.6: Esquema das partes principais de um sistema embebido.	78
Figura 3.7: Níveis de abstracção num sistema computacional típico.	81
Figura 3.8: Fertilização cruzada no co-projecto.	83
Figura 3.9: Exemplos de classes de componentes de <i>hardware</i>	84
Figura 3.10: Descrição de um registo em <i>C++</i>	84
Figura 3.11: Especialização de componentes de <i>hardware</i>	85
Figura 3.12: Descrição de um registo em <i>OO-VHDL</i>	85
Figura 3.13: Descrição de um contador em <i>OO-VHDL</i>	86
Figura 3.14: Características da metodologia de co-partição.	88
Figura 3.15: Momentos da partição <i>hardware/software</i>	89
Figura 3.16: Estratégias de escalonamento de processos.	90
Figura 3.17: Redução de lucros devido a atrasos no <i>time to market</i>	94
Figura 3.18: Tecnologia COTS.	94
Figura 3.19: Modelo do processo de desenvolvimento da abordagem orientada à concepção.	95
Figura 3.20: Modelo das tarefas de análise de requisitos e definição da arquitectura do sistema.	96
Figura 3.21: Modelo das tarefas de desenvolvimento do <i>hardware</i>	97
Figura 3.22: Prototipagem virtual no co-projecto <i>hardware/software</i>	99
Figura 4.1: Tecnologias de implementação de sistemas digitais.	116
Figura 4.2: Capacidades típicas dos FPLDs.	117
Figura 4.3: Comparação MPLDs vs. FPLDs.	119
Figura 4.4: Princípio da lógica reconfigurável.	121
Figura 4.5: Diagrama de blocos de um SPLD.	122
Figura 4.6: Programabilidade dos SPLDs.	123
Figura 4.7: Outros tipos de SPLDs.	123
Figura 4.8: Arquitectura CPLD típica.	124

Figura 4.9: Arquitetura da família <i>MACH 5</i> da <i>AMD</i>	125
Figura 4.10: Estrutura do bloco PAL da família <i>MACH 5</i>	125
Figura 4.11: Macrocélula e célula de I/O da família <i>MACH 5</i>	126
Figura 4.12: Cadeia <i>JTAG</i> para a programação de CPLDs ISP da família <i>MACH</i>	126
Figura 4.13: Arquitetura FPGA típica.	128
Figura 4.14: Implementação de lógica combinatória com LUTs de 5 entradas.	129
Figura 4.15: Arquitetura da família <i>Xc 6200</i> da <i>XILINX</i>	130
Figura 4.16: Célula lógica básica da família <i>Xc 6200</i> da <i>XILINX</i>	131
Figura 4.17: Funções lógicas da célula da família <i>Xc 6200</i> da <i>XILINX</i>	132
Figura 4.18: Reconfiguração de FPGAs.	136
Figura 4.19: Tipos de sistemas reconfiguráveis.	137
Figura 4.20: O sistema <i>RACE</i> como uma unidade externa de processamento reconfigurável.	138
Figura 4.21: O sistema <i>GARP</i> na implementação de um co-processador reconfigurável.	138
Figura 4.22: O sistema <i>CHIMAERA</i> como um processador com uma unidade funcional reconfigurável.	139
Figura 4.23: Topologia <i>Mesh</i>	140
Figura 4.24: Topologia <i>Crossbar</i> : A-D são <i>routing-only FPGAs</i> , W-Z são <i>logic-bearing FPGAs</i>	140
Figura 4.25: O sistema reconfigurável <i>SPLASH 2</i> (só estão representados 9 dos 16 FPGAs existentes).	141
Figura 4.26: O sistema reconfigurável <i>PAM</i>	142
Figura 4.27: Reconfiguração estática de <i>hardware</i>	143
Figura 4.28: Reconfiguração dinâmica de <i>hardware</i>	144
Figura 4.29: Reconfiguração dinâmica global de <i>hardware</i>	144
Figura 4.30: Reconfiguração dinâmica local de <i>hardware</i>	145
Figura 4.31: Modos de funcionamento de sistemas reconfiguráveis.	145
Figura 4.32: Comparação ASIPs vs. CCMs.	148
Figura 5.1: Modelo do processo de desenvolvimento da metodologia <i>INES</i> : macroprocesso e microprocesso da fase de análise.	164
Figura 5.2: Esquema resultante da primeira captura do ambiente das linhas HIDRO.	166
Figura 5.3: Analogia da casca da cebola.	166
Figura 5.4: Diagrama de contexto das linhas HIDRO.	167
Figura 5.5: Diagrama de casos de uso das linhas HIDRO.	169
Figura 5.6: Caso de uso 9 das linhas HIDRO.	170
Figura 5.7: Caso de uso 10a das linhas HIDRO, obtido por especialização.	171
Figura 5.8: Caso de uso 10b das linhas HIDRO, obtido por desagregação.	172

Figura 5.9: As três dimensões e os três tipos de objectos da metodologia <i>OOSE</i>	174
Figura 5.10: Diagrama de objectos das linhas HIDRO.	176
Figura 5.11: Diagrama de objectos colapsado das linhas HIDRO.	178
Figura 5.12: Sistema controlador e sistema controlado.	179
Figura 5.13: Planta genérica de uma linha de transporte superior – nó básico superior.	180
Figura 5.14: Planta genérica das três linhas de transporte superior – nó composto superior.	181
Figura 5.15: Um diagrama de sequência para as regras <i>rae-1</i> e <i>rae-2</i>	185
Figura 5.16: Um diagrama de cenário para a regra <i>rod-3</i>	186
Figura 5.17: Uma sequência de controlo.	193
Figura 5.18: Diagrama de objectos final de alto-nível das linhas HIDRO.	194
Figura 5.19: Filtragem do diagrama de objectos colapsado das linhas HIDRO.	195
Figura 5.20: Equivalências estruturais de arcos hierárquicos em <i>shobi-PN v2.0</i>	199
Figura 5.21: Rede de Petri <i>ContNCS3</i> – ciclo de vida do objecto <i>9.3.c controlador de nível 2</i>	200
Figura 5.22: Rede de Petri <i>Test_Sensor</i>	201
Figura 5.23: Outra rede de Petri <i>Test_Sensor</i>	203
Figura 5.24: Rede de Petri <i>ContNCES3I3</i> – outro ciclo de vida do objecto <i>9.3.c controlador de nível 2</i>	204
Figura 5.25: Rede de Petri <i>Send_Other</i> – controla as movimentações dos elevadores.	206
Figura 5.26: Mapeamento entre a realidade e o sistema, por continuidade dos modelos.	207
Figura 5.27: Tipos de herança.	209
Figura 5.28: Diagrama de classes dos controladores de nível 2 das linhas HIDRO.	210
Figura 5.29: Abordagem dos objectos dentro de redes de Petri - OIN.	211
Figura 5.30: Abordagem das redes dentro de objectos - NIO.	212
Figura 5.31: Abordagem mista do meta-modelo <i>shobi-PN v2.0</i>	213
Figura 5.32: Rede associada <i>Net2</i>	215
Figura 5.33: Código <i>OBLOG</i> da assinatura da classe abstracta <i>Parallel_State_Machine</i>	216
Figura 5.34: Código <i>OBLOG</i> da implementação da classe abstracta <i>Parallel_State_Machine</i>	217
Figura 5.35: Código <i>OBLOG</i> da assinatura da classe <i>Net2</i>	218
Figura 5.36: Código <i>OBLOG</i> de parte da implementação da classe <i>Net2</i>	219
Figura 5.37: Implementação em <i>OBLOG</i> das operações do tipo <i>state change methods</i>	220
Figura 5.38: Implementação em <i>OBLOG</i> das operações do tipo <i>even reaction oriented transition methods</i>	221
Figura 5.39: Implementação em <i>OBLOG</i> das operações do tipo <i>eventless transition methods</i>	222
Figura 5.40: Implementação em <i>OBLOG</i> das operações do tipo <i>state methods</i>	224
Figura 5.41: Implementação em <i>OBLOG</i> das operações <i>programmable-free hand-crafted</i>	225

Figura 5.42: Implementação em <i>OBLOG</i> das operações do tipo <i>sub-machine parameter event reaction</i> .	226
Figura 5.43: Implementação em <i>OBLOG</i> das operações do tipo <i>sub-machine return event reaction</i> .	227
Figura 5.44: Implementação em <i>OBLOG</i> do reset de sub-máquinas de estados.	228
Figura 5.45: Diagrama de classes da região de decomposição <i>Sensors</i> .	232
Figura 5.46: Diagrama de classes da região de decomposição <i>Actuators</i> .	233
Figura 5.47: Implementação em <i>OBLOG</i> das classes abstractas <i>Sensor</i> e <i>Actuator</i> .	234
Figura 5.48: Diagrama de classes da região de decomposição <i>Controller</i> .	235
Figura 5.49: Regiões de decomposição sugeridas pela metodologia <i>INES</i> .	235
Figura 5.50: Ambiente <i>JAVA</i> na animação de <i>shobi-PNs v2.0</i> .	237
Figura 5.51: Ambiente <i>ARENA</i> na simulação estatística do comportamento dos controladores.	237
Figura 6.1: Configuração organizacional I, no suporte a projectos de engenharia de computadores.	251
Figura 6.2: Configuração organizacional II, no suporte a projectos de engenharia de computadores.	252
Figura 6.3: Configuração organizacional III, no suporte a projectos de engenharia de computadores.	252
Figura 6.4: Configuração organizacional IV, no suporte a projectos de engenharia de computadores.	252
Figura 6.5: Configuração organizacional V, no suporte a projectos de engenharia de computadores.	253
Figura 6.6: Diagrama de contexto ao nível do processo.	255
Figura 6.7: Diagrama de interfaces dos <i>boards</i> .	255
Figura 6.8: Diagrama de casos de uso ao nível do processo.	257
Figura 6.9: Caso de uso 1. <i>desenvolver boards</i> .	259
Figura 6.10: Caso de uso 2. <i>programar boards</i> .	260
Figura 6.11: Caso de uso 5. <i>conceber solução</i> .	260
Figura 6.12: Diagrama de objectos ao nível do processo.	262
Figura 6.13: Modelação virtual nas ferramentas de nível 2 e 3 de (co-)projecto.	263
Figura 6.14: Ciclo de vida das soluções finais.	264
Figura 6.15: Macroprocesso global de desenvolvimento da metodologia <i>INES</i> .	266
Figura 6.16: Ambiente global EDA proposto pela metodologia <i>INES</i> .	268
Figura 6.17: Interligação de componentes.	271
Figura 6.18: Arquitectura alvo <i>CANit v4.0</i> .	275
Figura 6.19a: Diagrama de classes da arquitectura alvo <i>CANit v4.0</i> .	277
Figura 6.19b: Diagrama de classes da arquitectura alvo <i>CANit v4.0</i> (cont.).	278

Figura 6.20: Código C que implementa o construtor do objecto de baixo-nível <i>Beep</i>	280
Figura 6.21: Código <i>OBLOG</i> da classe de baixo-nível <i>Display</i>	281
Figura 6.22: Simulação em <i>JAVA</i> do objecto <i>display</i>	282
Figura 6.23: <i>OBLOG</i> vs. C.	282
Figura 6.24: Agregados de objectos de alto- e de baixo-nível.	283
Figura 6.25: Um FMOTS.	286
Figura 6.26: Diagrama de classes e de objectos de um FMOTS.	287
Figura 6.27: Código <i>OBLOG</i> do objecto de alto-nível principal de um FMOTS.	288
Figura 6.28: Mecanismo de recepção de mensagens.	290
Figura 6.29: Diagrama vectorial do fluxo de execução de um FMOTS.	290
Figura 6.30: Diagrama pictórico do padrão <i>MVC</i>	292
Figura 6.31: Diagrama de classes do padrão <i>MVC</i> , em notação <i>UML</i>	292
Figura 6.32: Diagrama de sequência de um cenário do padrão <i>MVC</i> , em notação <i>UML</i>	293
Figura 6.33: Diagrama pictórico do padrão <i>Multi-Level ICIS</i>	294
Figura 6.34: Topologia tipo das soluções finais.	295
Figura 6.35: Diagrama de disposição da arquitectura de uma solução final (<i>ICIS</i>).	296
Figura 6.36: Diagrama de disposição da arquitectura do software de uma solução final.	297
Figura 6.37: Porção de código <i>LabVIEW</i> de uma solução final.	297
Figura 6.38: Diagrama de disposição da arquitectura do software de um FMOTS.	298
Figura 6.39: Diagrama de classes do padrão <i>Multi-Level ICIS</i> , em notação <i>UML</i>	298
Figura 6.40: Diagrama de sequência de um cenário do padrão <i>Multi-Level ICIS</i> , em notação <i>UML</i>	299
Figura 6.41: Diferentes níveis de generalização de FMOTSs.	300
Figura 6.42: Nível superior de uma <i>shobi-PN v2.0</i> geratriz do FMOTS da folha 3.1.3.1.	303
Figura 6.43: Nível superior de uma <i>shobi-PN v2.0</i> para os teares <i>LONATI 409/421</i>	304
Figura 6.44: Diagrama FIDL.	306
Figura 6.45: Esquemático (parcial) das interfaces do FMOTS 3.1.3.1.	307
Figura 6.46: Diagrama FIDL do atributo <i>startOut_2</i>	311
Figura 6.47: Diagrama FIDL do atributo <i>startOut_1</i>	312
Figura 6.48: Diagrama FIDL do evento <i>actMachineError</i>	313
Figura 6.49: Diagrama FIDL do atributo <i>efficiency</i>	314
Figura 7.1: Evolução funcional dos RTOSs.	329
Figura 7.2: Solução sequencial da arquitectura alvo <i>CANit-FPGA v1</i>	334
Figura 7.3: Solução concorrente da arquitectura alvo <i>CANit-FPGA v1</i>	335

Figura 7.4: Solução intermédia da arquitectura alvo <i>CANit-FPGA v1</i>	335
Figura 7.5: Solução “ <i>transceiver</i> ” para a arquitectura alvo <i>CANit-FPGA v1</i> com acesso a redes <i>ETHERNET</i>	336
Figura 7.6: Solução “ <i>transceiver + controlador MAC</i> ” para a arquitectura alvo <i>CANit-FPGA v1</i> com acesso a redes <i>ETHERNET</i>	337
Figura 7.7: Modelo de implementação para execução evolutiva e paralela dos algoritmos.	338
Figura 7.8: Diagrama de alto-nível de meta-classes <i>OBLOG</i>	341
Figura 7.9: Meta-classe <i>MClass</i>	341
Figura 7.10: Meta-classe <i>MOperation</i>	342
Figura 7.11: Regra <i>GetIfBody</i> escrita em <i>script RDL</i> para gerar expressões condicionais em <i>C</i>	343
Figura 7.12: Operação <i>stopHandling</i> em linguagem <i>OBLOG</i>	344
Figura 7.13: Função <i>Sitaf_stopHandling</i> em linguagem <i>C</i>	344
Figura 7.14: Escalonamento global de um FMOTS.	345
Figura 7.15a: Animação de um FMOTS no ambiente <i>JAVA</i> (criação das instâncias).	347
Figura 7.15b: Animação de um FMOTS no ambiente <i>JAVA</i> (troca de mensagens).	347
Figura 7.16: Modelo virtual do FMOTS 3.1.3.1. e “unbundle” de alguns atributos.	350
Figura 7.17: Topologia genérica da rede de comunicações de um ICIS.	351
Figura 7.18: Algumas paletas do menu da VAL (<i>biblioteca virtual automation</i>).	352
Figura 7.19: Declaração de <i>CANits</i> e <i>CANios</i>	353
Figura 7.20: Iniciação de serviços de comunicação.	353
Figura 7.21: Parametização de FMOTSs.	354
Figura 7.22: Interligação ICIS - MIS.	354
Figura 7.23: Início de um projecto de <i>virtual automation</i> , em <i>LabVIEW</i>	355
Figura 7.24: Topologia da versão protótipo da solução final do SITAF.	356
Figura 7.25: Interface gráfica do <i>PC-VAP</i> na re-parametização dos <i>CANits</i> e <i>CANios</i>	357
Figura 7.26: Interface gráfica do <i>PC Produção</i> do projecto SITAF.	358
Figura 7.27: Interface gráfica do <i>PC Manutenção</i> do projecto SITAF.	358
Figura 7.28: Vista de um relatório gerado automaticamente pelo <i>CRYSTAL REPORT</i> (projecto SITAF).	359
Figura 7.29: Interface gráfica do <i>PC Direcção</i> do projecto SITAF.	359

Índice de Tabelas

Tabela 1.1: Divergências entre os MISs e os ICISs.	15
Tabela 1.2: Estrutura da tese.	20
Tabela 2.1: Desenvolvimento tecnológico básico dos CBSs tempo-real.	36
Tabela 3.1: Níveis de abstracção e domínios de representação de sistemas digitais.	81
Tabela 3.2: Resumo das características das abordagens ao co-projecto.	105
Tabela 4.1: Comparação MPLDs vs. FPLDs.	120
Tabela 4.2: Comparação de vários CPLDs ISP.	127
Tabela 4.3: Comparação de vários FPGAs ISP.	132
Tabela 4.4: Tecnologias de programação dos FPLDs.	134
Tabela 5.1: Excerto da lista de sensores existentes nas linhas HIDRO.	182
Tabela 5.2: Interpretações de redes de Petri.	189
Tabela 5.3: Conjunto dos arcos generalizados do meta-modelo <i>shobi-PN v2.0</i>	197
Tabela 6.1: Glossário dos casos de uso ao nível do processo.	256
Tabela 6.2: Evolução no projecto de CBSs.	267
Tabela 6.3: Parâmetros dos equipamentos do FMOTS 3.1.3.1.	305

Acrónimos e Abreviaturas

“You have to study a great deal to know a little”

Charles de Secondat, Baron de Montesquieu

5 T's	<i>timelines, tasks, tools, technology, talent</i>
ACID	<i>atomicity, consistency, isolation, durability</i>
ADC	<i>analog/digital converter</i>
ADT	<i>abstract data type</i>
ALU	<i>arithmetic and logic unit</i>
AMT	<i>asynchronous macro-transition</i>
API	<i>application program interface</i>
ASIC	<i>application-specific integrated circuit</i>
ASIP	<i>application-specific instruction-set processor</i>
BD	<i>block diagram</i>
bit	<i>binary digit</i>
BJT	<i>bipolar junction transistor</i>
BLIF	<i>Berkeley logic interchange format</i>
BLIF-MV	<i>multi-valued Berkeley logic interchange format</i>
BPM/E	<i>business process management/engineering</i>
CAD	<i>computer aided design</i>
CAE	<i>computer aided engineering</i>
CAM	<i>computer aided manufacturing</i>
CAN	<i>controller area network</i>
CANit	<i>CAN industrial terminal</i>
CASE	<i>computer aided software engineering</i>
CBD	<i>component-based development</i>
CBS	<i>computer-based system</i>
CCD	<i>component-connectivity diagram</i>
CDFG	<i>control/data flow graph</i>
CFG	<i>control flow graph</i>

CFSM	<i>co-design finite-state machine</i>
CIA	<i>CAN in Automation International Users and Manufacturers Group e. V.</i>
CIM	<i>computer integrated manufacturing</i>
CLB	<i>configurable logic block</i>
CMOS	<i>complementary metal-oxide semiconductor</i>
CNC	<i>computer numerically controlled</i>
CODES	<i>concurrent design environment</i>
COM	<i>component object model</i>
CORBA	<i>common object request broker architecture</i>
COSYMA	<i>cosynthesis for embedded micro architectures</i>
COTS	<i>component-off-the-self</i>
CPLD	<i>complex programmable logic device</i>
CPU	<i>central processing unit</i>
CSMA/CD	<i>carrier sense multiple access with collision detected</i>
CTR	<i>compile-time reconfiguration</i>
DCOM	<i>distributed COM</i>
DFD	<i>data flow diagram</i>
DFG	<i>data flow graph</i>
DLL	<i>dynamic link library</i>
DSP	<i>digital signal processor</i>
ECAD	<i>electronic computer aided design</i>
ECL	<i>emitter-coupled logic</i>
EDA	<i>electronic design automation</i>
EDF	<i>earliest deadline first</i>
EDIF	<i>electronic design information interchange format</i>
EEPROM	<i>electrically EPROM (=E2PROM)</i>
EFSM	<i>extended finite-state machine</i>
EPLD	<i>electrically programmable logic device</i>
EPROM	<i>erasable programmable read-only memory</i>
ERD	<i>entity-relationship diagram</i>
ERP	<i>enterprise resources planning</i>
ESA	<i>European Space Agency</i>
ESG	<i>extended syntax graph</i>
ETPN	<i>extended timed Petri net</i>
FC	<i>flowchart</i>
F-CCM	<i>FPGA-based custom computing machine</i>
FIDL	<i>FMOTSS interface definition language</i>
FMOTS	<i>functional-module-off-the-shelf</i>
FMS	<i>flexible manufacturing systems</i>

FPCB	<i>field-programmable circuit board</i>
FPGA	<i>field programmable gate-array</i>
FPI	<i>field-programmable interconnect</i>
FPIC	<i>field-programmable interconnect component</i>
FPID	<i>field-programmable interconnect device</i>
FPLD	<i>field-programmable logic device</i>
FSA	<i>finite-state automata</i>
FSM	<i>finite-state machine</i>
FSMD	<i>finite-state machine with data path</i>
GAL	<i>generic array logic</i>
GAS	<i>generalized arc set</i>
GSM	<i>global system for mobile communications</i>
GUI	<i>graphical user interface</i>
HCFSM	<i>hierarchical concurrent finite-state machine</i>
HCPLD	<i>high-capacity programmable logic device</i>
HDL	<i>hardware description language</i>
HLL	<i>high-level language</i>
HLS	<i>high-level synthesis</i>
HMI	<i>human-machine interface</i>
HOOD	<i>hierarchical object-oriented design</i>
HPC	<i>high performance computer</i>
HTML	<i>hypertext mark-up language</i>
HTTP	<i>hypertext transfer protocol</i>
I&D	<i>investigação e desenvolvimento</i>
I/O	<i>input and output</i>
ICIS	<i>industrial control-based information system</i>
IEC	<i>International Electrotechnical Commission</i>
IEEE	<i>Institute of Electrical and Electronics Engineers</i>
ILP	<i>instruction-level parallelism</i>
INES	<i>integrated environment for embedded systems design</i>
INSYDE	<i>integrated system design</i>
IOB	<i>input/output block</i>
IP	<i>internet protocol</i>
ISO	<i>International Organisation for Standardisation</i>
ISP	<i>instruction-set processor</i>
ISP	<i>in-system programming</i>
JEDEC	<i>Joint Electron Device Engineering Council</i>
JSD	<i>Jackson's structured diagram</i>
JTAG	<i>Joint Test Action Group</i>

JVM	<i>JAVA virtual machine</i>
KISS	<i>keep it simple, stupid!</i>
LCA	<i>logic cell array</i>
LED	<i>light emitting diode</i>
LLC	<i>logic link control</i>
LSI	<i>large-scale integration</i>
LUT	<i>lookup table</i>
LYCOS	<i>Lynby co-synthesis</i>
MAC	<i>medium access control</i>
MAP	<i>manufacturing automation protocol</i>
MCSE	<i>methodology for electronic systems design</i>
MIS	<i>management information system</i>
MMS	<i>manufacturing message specification</i>
MOOSE	<i>model-based object oriented systems engineering</i>
MOSFET	<i>metal-oxide semiconductor field-effect transistor</i>
MPGA	<i>mask programmable gate array</i>
MPLD	<i>mask programmable logic device</i>
MSI	<i>medium-scale integration</i>
NASA	<i>NATIONAL AERONAUTICS & SPACE ADMINISTRATION</i>
NIO	<i>nets inside objects</i>
NRE	<i>nonrecurring engineering costs</i>
NUMA	<i>non-uniform memory access</i>
OBLOG	<i>object logic</i>
OCL	<i>object constraint language</i>
OIN	<i>objects inside nets</i>
OLE	<i>object linking and embedding</i>
OMG	<i>object management group</i>
OMT	<i>object modeling technique</i>
OO	<i>object-orientation</i>
OOA	<i>object-oriented analysis</i>
OOAD	<i>object-oriented analysis and design</i>
OOD	<i>object-oriented design</i>
OOIE	<i>object-oriented information engineering</i>
OORT	<i>object-oriented real-time techniques</i>
OOSE	<i>object-oriented software engineering</i>
OPC	<i>OLE for process control</i>
ORB	<i>object request broker</i>
PAL	<i>programmable array logic</i>
PCB	<i>printed circuit board</i>

PDA	<i>personal digital assistant</i>
PERC	<i>portable executive reliable control</i>
PL	<i>programming language</i>
PLA	<i>programmable logic array</i>
PLC	<i>programmable logic controller</i>
PN	<i>Petri net</i>
POS	<i>plant operations system</i>
POSIX	<i>portable operating system interface for computer environments</i>
PR/T-NET	<i>predicate transition net</i>
PRAM	<i>parallel random access machine</i>
PROM	<i>programmable read-only memory</i>
PSM	<i>program-state machine</i>
RDL	<i>rule definition language</i>
RMS	<i>rate monotonic scheduling</i>
ROOM	<i>real-time object-oriented modeling</i>
RPC	<i>remote procedure call</i>
RTEUML	<i>real-time extensions to unified modeling language</i>
RTJEG	<i>Real-Time for JAVA Experts Group</i>
RTL	<i>register transfer level</i>
RTOS	<i>real-time operating system</i>
RTR	<i>run-time reconfiguration</i>
RTSA	<i>real-time structured analysis</i>
RTSJ	<i>real-time specification for JAVA</i>
RTUML	<i>real-time unified modeling language</i>
SC	<i>structure chart</i>
SCADA	<i>supervision, control, automation and data acquisition</i>
SDRTS	<i>structured development for real-time systems</i>
SHIFT	<i>software hardware interchange format</i>
shobi-PN	<i>synchronous hierarchical object-oriented and interpreted Petri net</i>
SIPN	<i>synchronous and interpreted Petri net</i>
SLDL	<i>system-level design language</i>
SMS	<i>short message service</i>
SMT	<i>synchronous macro-transition</i>
SMTP	<i>simple mail transfer protocol</i>
SOFHIA	<i>software for hierarchical architectures</i>
SOMA	<i>semantic object modeling approach</i>
SPLD	<i>simple programmable logic device</i>
SQL	<i>structured query language</i>
SRAM	<i>static random-access memory</i>

SRTSS	<i>strategies for real-time system specification</i>
SSI	<i>small-scale integration</i>
STD	<i>state transition diagram</i>
TAP	<i>test access port</i>
TCP	<i>transmission control protocol</i>
TOP	<i>technical and office protocol</i>
TOSCA	<i>tools for system co-design automation</i>
TTL	<i>transistor-transistor logic</i>
UML	<i>unified modeling language</i>
UMLRT	<i>UML for real-time</i>
VAB-CCM	<i>VLSI accelerator-based custom computing machine</i>
VAL	<i>virtual automation library</i>
VAP	<i>virtual automation protocol</i>
VASch	<i>virtual automation scheduler</i>
VHDL	<i>VHSIC HDL (very high speed integrated circuit hardware description language)</i>
VHSIC	<i>very high speed integrated circuit</i>
VI	<i>virtual instrument</i>
VLSI	<i>very-large-scale integration</i>
WAP	<i>wireless application protocol</i>
WML	<i>wireless mark-up language</i>
WCET	<i>worst-case execution time</i>
WORA	<i>write once, run anywhere</i>

Prefácio

“O grau de doutor comprova a realização de uma contribuição inovadora e original para o progresso do conhecimento, um alto nível cultural numa determinada área do conhecimento e a aptidão para realizar trabalho científico independente.”

Art.º 17º, Cap.º III, Dec.-Lei n.º 216/92, de 13 de Outubro, Ministério da Educação

Enquadramento

A estimativa recente de que nos países desenvolvidos (classificação da responsabilidade da OCDE) existe mais do que uma dezena de sistemas embebidos por cada habitante, leva-me a pensar na enorme quantidade de profissionais que se dedica actualmente ao projecto (desenvolvimento/programação) desses sistemas, tendo em conta que ainda não é usual os *curricula* das licenciaturas (em engenharia electrotécnica, engenharia de computadores e engenharia informática) disponibilizarem disciplinas naquela área, bem como o facto de não existir uma grande oferta bibliográfica acerca do projecto daquele tipo de sistemas.

Neste cenário pouco amistoso para quem acaba por se dedicar profissionalmente à área dos sistemas embebidos, tive a sorte de lidar com aqueles sistemas em duas realidades estruturantes para a minha formação como engenheiro. Na primeira, ainda como estagiário de licenciatura na ENSEA em Cergy/Paris, tive a oportunidade de verificar a importância dos sistemas embebidos para a electrónica de consumo, uma vez que estive envolvido no desenvolvimento de uma unidade de processamento de voz em tempo-real para incorporar (embeber) num produto de um dos maiores fabricantes franceses de sistemas de áudio profissional, a *BARTHE*. A segunda, já como engenheiro de desenvolvimento da TISEP na Maia/Porto, fui confrontado com a realidade industrial dos sistemas embebidos, uma vez que fui “obrigado” a projectar alguns para supervisionarem, monitorizarem e controlarem equipamentos e processos produtivos (um dos quais deu, agradavelmente, origem ao registo de uma patente).

Estas duas experiências pessoais acabaram por marcar profundamente este doutoramento, pelo facto de me terem sensibilizado para a importância da fertilização cruzada entre a Universidade e a Indústria na execução das actividades de investigação: (1) no sentido *Indústria* → *Universidade* com o objectivo de sensibilizar e influenciar a investigação na direcção das necessidades reais das pessoas; (2) no sentido *Universidade* → *Indústria* com o objectivo de introduzir nos processos, sistemas e produtos reais as inovações científicas e

tecnológicas, desenvolvidas no seio de um ambiente protegido e despreocupado das pressões produtivas e comerciais sempre presentes nos ambientes empresariais. A defesa da necessidade desta fertilização cruzada é, aliás, extrapolada para a própria actividade de docência universitária pelo Professor Doutor Eng.º Júlio Barreiros Martins, aquando da sua jubilação na *UNIVERSIDADE DO MINHO* em Julho de 2000 (da qual foi publicado um excerto pela revista *Ingenium*, em Setembro de 2000, com o título *Investigação Fundamental e Investigação Aplicada*), quando refere que:

“... o professor universitário, para verdadeiramente o ser, tem de estar ligado à investigação aplicada e portanto no acompanhamento das grandes obras e seus projectos e/ou na resolução de problemas que a respeito das obras reais sempre se colocam.”.

Outra influência marcante nesta tese consiste no facto da minha formação de base (licenciatura em Engenharia Electrotécnica e de Computadores) me ter fornecido uma visão clássica da disciplina de projecto, em que a abordagem é solidamente fundamentada nos princípios científicos e tecnológicos comprovadamente aceites como válidos para basear todas as tomadas de decisão conducentes à implementação sustentada do sistema final em projecto. Esta realidade metodológica ainda não é totalmente possível no projecto de sistemas embebidos, tal como é magnificamente referido pelo Prof. Doutor Eng.º José Nuno Oliveira na sua “sebenta” da disciplina de *Especificação e Desenvolvimento Formal de Software* do Mestrado em Informática da *UNIVERSIDADE DO MINHO* (disciplina que eu tive o enorme prazer de frequentar há alguns anos atrás) a propósito da reificação de *software*:

“Assim, todos os “truques” de programação introduzidos numa implementação para aumentar a sua eficiência são, por inerência, detalhes irrelevantes e sem significado ao nível da especificação de que se partiu. Em suma, o salto “epistemológico” entre especificações e implementações está longe de ser “suave” e é a principal preocupação da chamada tecnologia da reificação (ou refinamento), um ramo relativamente recente da engenharia da programação de computadores...”.

Perante este cenário de influência multifactorial (no que diz respeito à minha motivação pessoal para ter tratado, nesta tese, o tema do projecto de sistemas embebidos), este doutoramento foi desenvolvido, numa continuação explícita e intencional dos trabalhos de investigação iniciados no mestrado, como uma “cruzada” pessoal, com o objectivo de contribuir para que a disciplina de projecto em engenharia de computadores venha a ser considerada, um dia, uma disciplina clássica de projecto, pelas vindouras especialidades de engenharia que já se adivinham neste, quase, início de século XXI.

Agradecimentos

Esta tese de doutoramento é o resultado de cerca de três anos de trabalho intensivo que não poderia ter sido desenvolvido e concluído sem a inevitável colaboração de várias pessoas e entidades, às quais gostaria de deixar aqui expresso o meu reconhecimento público.

Em primeiro lugar, pretendo agradecer sinceramente ao meu orientador científico, o *Prof. Doutor Eng.º Henrique Santos*, do departamento de Sistemas de Informação da *ESCOLA DE ENGENHARIA* da *UNIVERSIDADE DO MINHO*, o excepcional clima de abertura e diálogo que sempre incutiu nas inúmeras conversas e discussões que travámos ao longo deste anos. Este clima de informalidade, cordialidade e cooperação revelaram-se imprescindíveis numa tarefa que exige uma elevadíssima perseverança e resistência psicológica, como são a prossecução dos trabalhos de I&D subjacentes a um doutoramento, bem como a escrita propriamente dita do documento que materializa a tese, tantas são as adversidades que um doutorando português enfrenta ao longo do seu percurso para atingir a meta do grau académico desejado.

Adicionalmente, queria agradecer-lhe as conversas frutíferas que me proporcionou e os muitos comentários, críticas e sugestões que valorizaram enormemente esta tese. Obrigado por ter sabido encontrar o equilíbrio entre o *orientar* e o *criar espaço para aprender!*

Tenho de deixar expressa uma palavra de profundo agradecimento ao meu colega (recém Prof. Doutor) *Eng.º João Miguel Fernandes*, do departamento de Informática da *ESCOLA DE ENGENHARIA da UNIVERSIDADE DO MINHO*, pelo facto de me ter acompanhado nesta “cruzada” pessoal, desde que, em Julho de 1995, me encontrei com ele pela primeira vez nos, antigos, corredores do departamento de Informática. Desde então, o nosso percurso tem sido trilhado em conjunto, num equilíbrio perfeito entre a necessária satisfação legítima das aspirações pessoais e o desenvolvimento de esforços consertados na persecução de metas mais ambiciosas, somente alcançáveis através de uma adequada rentabilização das sinergias de grupo, que se traduziram, nomeadamente, nas actividades de co-coordenação operacional dos três projectos que estão na base deste doutoramento, para além das variadíssimas publicações científicas que juntos temos escrito. Não tenho quaisquer dúvidas em afirmar que sem a sua participação aqueles três projectos teriam seguido um rumo menos interessante. Agradeço-lhe também o tempo dispensado na revisão do texto desta tese.

Ao *Eng.º Braz Costa*, director-geral do *CITEVE* e director executivo da iniciativa *SIPROFIT*, ao *Eng.º Eduardo Pinto*, director operacional do *IDITE-Minho*, e ao *Eng.º Adelino Silva*, chefe do projecto *INFRACOM*, por terem acreditado que as minhas “teorias” poderiam ser aplicadas à realidade industrial portuguesa.

Ao *Eng.º José Cunha*, director de produção da *BLAUPUNKT AUTO-RÁDIOS PORTUGAL*, e ao *Dr. António Falcão*, director-geral da *TÊXTIL A. FALCÃO S.A.*, por terem promovido e cultivado uma aproximação exemplar entre a Universidade e a Indústria.

Ao *Dr. Luís Andrade* e ao *Dr. João Gouveia*, directores da *OBLOG SOFTWARE S.A.*, e ao *Eng.º Pedro Alves*, director da *NATIONAL INSTRUMENTS* em Portugal, pelas facilidades concedidas na utilização das ferramentas de *CASE (OBLOG EDITOR e OBLOG GENERATOR)* e de *CAE (LabVIEW)* que representam e pela prestável ajuda no esclarecimento das diversas dúvidas de carácter técnico que foram surgindo, ao longo destes anos, relativamente à utilização das referidas ferramentas.

Ao *Eng.º Jorge Cruz*, ao *Eng.º Francisco Duarte* e ao *Eng.º Fernando Barbosa*, os três quadros técnicos superiores da *BLAUPUNKT AUTO-RÁDIOS PORTUGAL*, pela forma dedicada e profissional como trabalharam no projecto de *Diagnóstico e Optimização do Sistema de Controlo das Linhas HIDRO*. Agradeço, igualmente, aos colegas *Prof. Doutor Eng.º Guilherme Pereira* e *Eng.º Luís Silva Dias*, do departamento de Produção e Sistemas da *ESCOLA DE ENGENHARIA da UNIVERSIDADE DO MINHO*, a preciosa colaboração prestada na simulação estatística do controlador especificado no âmbito do projecto acima referido. Com esta equipa materializou-se a primeira pedra basilar do meu doutoramento, a metodologia de especificação.

À equipa de desenvolvimento do projecto *Virtual Automation* (nas pessoas da *Celeste Pinto*, da *Dr.ª Paula Monteiro*, do *Eng.º Vitor Macedo*, do *Eng.º Sérgio Dias*, do *Eng.º Manuel Carvalho*, do *Eng.º Nuno Rodrigues* e do *Luís Costa*) queria exprimir os meus sinceros agradecimentos pela excepcional dedicação a um projecto que, devido à sua natureza, tanto trabalho e esforço tem exigido, de todos, para que a equipa, como um todo, atinja a tão almejada “luz ao fundo do túnel”. Com esta equipa materializou-se a segunda pedra basilar do meu doutoramento, a metodologia de concepção.

Aos diversos alunos das licenciaturas em Matemática e Ciências da Computação, em Engenharia de Sistemas e Informática e em Engenharia Electrónica Industrial da *UNIVERSIDADE DO MINHO* que comigo colaboraram (*Ana Alves, Celeste Pinto, Paula Monteiro, Filipe Pereira, Hugo Paredes, Manuel Carvalho, Luís Costa, Rui Pimenta e Gonçalo Areias*), através da realização de projectos e estágios de final de curso, permitindo adiantar algumas das actividades realizadas nesta tese.

Do ponto de vista institucional, queria agradecer à *UNIVERSIDADE DO MINHO* (na pessoa do seu actual reitor, o *Professor Doutor Eng.º Chainho Pereira*), à *ESCOLA DE ENGENHARIA* (na pessoa do seu actual presidente, o *Professor Doutor Eng.º Guimarães Rodrigues*) e ao *CENTRO DE INVESTIGAÇÃO ALGORITMI* (na pessoa do seu actual director, o *Prof. Doutor Eng.º João Monteiro*) por terem disponibilizado as condições possíveis para suportar, dentro do exequível, a prossecução deste trabalho de doutoramento.

Do ponto de vista departamental, queria agradecer ao departamento de Informática (na pessoa do seu actual director, o *Prof. Doutor Eng.º Francisco Moura*), por, em Julho de 1997, ter apoiado a submissão, ao Conselho Científico da Escola, da minha proposta de plano de doutoramento ainda como docente daquele departamento, e ao departamento de Sistemas de Informação, o meu actual departamento, (nas pessoas dos seus actuais directores, o *Professor Doutor Eng.º Altamiro Machado* e o *Prof. Doutor Eng.º João Álvaro Carvalho*) por, desde a sua pré-génese ainda como NDIG (núcleo do departamento de Informática em Guimarães), ter acolhido com cordialidade e respeito o trabalho científico que tenho desenvolvido. Gostaria de agradecer especialmente ao *Professor Doutor Eng.º Altamiro Machado* pela força e incentivo que sempre me deu no “tratamento” das variantes *não convencionais* dos sistemas de informação.

Ao *ME/PRODEP* e à *FCT/PRAXIS*, pelo apoio financeiro que, directa ou indirectamente, concederam a esta tese, nomeadamente, na aquisição de material bibliográfico e no suporte às deslocações para apresentação de comunicações científicas em conferências e simpósios nacionais e internacionais na divulgação dos resultados deste doutoramento.

À *Sofia* e à pequena *Inês* agradeço toda a compreensão demonstrada ao longo destes cerca de 3 anos aquando da minha dupla actividade de “engenheiro de dia” e de “investigador à noite” que a tantos sacrifícios familiares obrigou. Assim, agradeço-lhes, sobretudo, todos aqueles serões, fins de semana, férias, horas, minutos, segundos ... que me “emprestaram” e que lhes pertenciam, mas que sem os quais não teria sido possível terminar este meu projecto pessoal e profissional tão rapidamente. Comprometo-me a devolver-lhes, com juro, o tempo perdido! Adicionalmente, agradeço à *Sofia* a ajuda e o tempo dispensados na revisão do texto desta tese.

Por último, não posso esquecer os meus *pais* pela forma dedicada como sempre me inculiram rigor no estudo e gosto pela investigação (que vieram a revelar-se cruciais para a minha carreira académica), nem a minha irmã *Rita* pelo seu incondicional apoio...

A todos muito obrigado !

Ricardo Jorge Silvério de Magalhães Machado

Outubro de 2000

Constituição do Júri*

Presidente	<i>Prof. Doutor Eng.º</i> Carlos Bernardo	Prof. Catedrático	Vice-Reitor da Universidade do Minho
Arguente	<i>Prof. Doutor Eng.º</i> João Gabriel Monteiro de Carvalho e Silva	Prof. Associado com Agregação	Dept. de Eng.ª Informática Faculdade de Ciências e Tecnologia Universidade de Coimbra
Arguente	<i>Prof. Doutor Eng.º</i> Adolfo Sanchez Steiger Garção	Prof. Catedrático	Dept. de Eng.ª Electrotécnica Faculdade de Ciências e Tecnologia Universidade Nova de Lisboa
Vogal	<i>Prof. Doutor Eng.º</i> José Alfredo Ribeiro da Silva Matos	Prof. Catedrático	Dept. de Eng.ª Electrotécnica e de Computadores Faculdade de Engenharia Universidade do Porto
Vogal	<i>Prof. Doutor Eng.º</i> Altamiro Barbosa Machado	Prof. Catedrático	Dept. de Sistemas de Informação Escola de Engenharia Universidade do Minho
Vogal	<i>Prof. Doutor Eng.º</i> Carlos Alberto Caridade Monteiro e Couto	Prof. Catedrático	Dept. de Electrónica Industrial Escola de Engenharia Universidade do Minho
Vogal	<i>Prof. Doutor Eng.º</i> Alberto José Gonçalves de Carvalho Proença	Prof. Catedrático	Dept. de Informática Escola de Engenharia Universidade do Minho
Supervisor	<i>Prof. Doutor Eng.º</i> Henrique Manuel Dinis dos Santos	Prof. Associado	Dept. de Sistemas de Informação Escola de Engenharia Universidade do Minho

(*) As provas públicas de doutoramento foram realizadas no dia 20 de Março de 2001, na sala do Senado, na Reitoria da Universidade do Minho.